Advant® OCS
with Master software

Advant Station™ 100 Series Information Management Station

User's Guide

Use of **DANGER**, **WARNING**, **CAUTION**, and **NOTE**

This publication includes, **DANGER**, **WARNING**, **CAUTION**, and **NOTE** information where appropriate to point out safety related or other important information.

DANGER	Hazards which could result in severe personal injury or death
WARNING	Hazards which could result in personal injury
CAUTION	Hazards which could result in equipment or property damage
NOTE	Alerts user to pertinent facts and conditions

Although **DANGER** and **WARNING** hazards are related to personal injury, and **CAUTION** hazards are associated with equipment or property damage, it should be understood that operation of damaged equipment could, under certain operational conditions, result in degraded process performance leading to personal injury or death. Therefore, comply fully with all **DANGER**, **WARNING**, and **CAUTION** notices.

TRADEMARKS

Advant, Advant Station, MOD 300 and Taylor are trademarks, and Master is a registered trademark of Asea Brown Boveri

Lotus 1-2-3 and Symphony are registered trademarks of Lotus Development Corporation

Microsoft, MS, and MS-DOS are registered trademarks, and Windows is a trademark of Microsoft Corporation

Quatro Pro is a trademark of Borland International Inc.

PC-DOS is a registered trademark of International Business Machines Corporation

3Com is a registered trademark of 3Com Corporation

Compaq and Deskpro are registered trademarks of Compaq Computer Corporation

NOTICE

The information in this document is subject to change without notice and should not be construed as a commitment by ABB Industrial Systems AB. ABB Industrial Systems AB assumes no responsibility for any errors that may appear in this document.

In no event shall ABB Industrial Systems AB be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall ABB Industrial Systems AB be liable for incidental or consequential damages arising from use of any software or hardware described in this document.

This document and parts thereof must not be reproduced or copied without ABB Industrial Systems AB’s written permission, and the contents thereof must not be imparted to a third party nor be used for any unauthorized purpose.

The software described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license.

Copyright © ABB Industrial Systems AB 1996

TABLE OF CONTENTS

Chapter 1 - Introduction

1.1 General Information 1-1
1.2 Manual Organization 1-2
1.3 Related Documentation 1-4
1.4 Release History 1-4
1.5 Terminology 1-5
1.6 Equipment Requirements 1-6
1.7 Product Overview 1-7

Chapter 2 - Installation

2.1 Site-Planning Environment 2-1
  2.1.1 SuperView 100Configuration 2-1
2.2 Setup 2-3
  2.2.1 Hardware 2-3
  2.2.2 Software 2-3
2.3 Shutdown Procedures 2-9
2.4 Start-Up Procedures 2-9
2.5 Product Verification 2-10

Chapter 3 - Configuration/Application Building

3.1 Application Procedures 3-1
  3.1.1 Report Production with Spreadsheet Software 3-2
    3.1.1.1 Defining a Report 3-3
    3.1.1.2 Requesting Report Data 3-6
    3.1.1.3 Creating Reports 3-7
    3.1.1.4 Sending Data to the ABB Automation System 3-10
  3.1.2 Report Scheduling with SuperView 100 Batch Files 3-11
  3.1.3 Using Dynamic Data Exchange (DDE) 3-12
  3.1.4 The Remote File System (RFS) Server 3-13
  3.1.5 The RFS File Transfer Library 3-13
3.2 Design Considerations 3-14
  3.2.1 Report Planning and Design 3-14
  3.2.2 Report Generation Strategies 3-15
3.3 Capacity and Performance 3-20
  3.3.1 SuperView 100Report Software 3-20
  3.3.2 The Remote File System 3-22
  3.3.3 Primary Memory 3-22
  3.3.4 Disk Storage 3-24

CONTENTS (continued)

3.4 Software Setup 3-25
3.5 Tutorials 3-25
    3.5.1 Report Production with Spreadsheet Software 3-26
    3.5.2 Using Batch Files For Report Scheduling 3-32
    3.5.3 Using DDE to Access Process Data 3-34
3.6 Menus 3-38

Chapter 4 - Runtime Operation

4.1 Operating Overview 4-1
4.2 Product Operation 4-1
    4.2.1 SuperView 100 Report Software 4-1
        4.2.1.1 The Windows Interface 4-2
        4.2.1.2 The Command Line Interface 4-6
        4.2.1.3 The DDE Interface 4-7
    4.2.2 The RFS Server 4-8
    4.2.3 The RFS File Transfer Library 4-9
4.3 Operating Instructions 4-10
    4.3.1 Batch File Execution 4-10
    4.3.2 Report Execution 4-12
    4.3.3 Symbol Resolution 4-14
    4.3.4 GCOM Configuration 4-15
    4.3.5 GCOM Status 4-16
    4.3.6 DDE Configuration 4-16
    4.3.7 GCOM Network Connection 4-17
    4.3.8 Help 4-17
    4.3.9 Current SuperView 100 Version 4-18
    4.3.10 RFS Configuration 4-20

Chapter 5 - Maintenance

5.1 Preventive Maintenance 5-1
5.2 Hardware Indicators 5-1
5.3 Error Messages 5-1
    5.3.1 Error Labels 5-1
5.4 Fault-Finding and User Repair 5-3

Appendix A - Network and Node Configuration Rules

A.1 General A-1

CONTENTS (continued)

Appendix B - Files Created by SuperView 100

B.1 General....B-1
B.2 Setup File....B-2
B.3 Symbol Files....B-4
B.4 TTD Symbol Files....B-5
B.5 RFS Description Files....B-6
B.6 RFS Configuration File....B-7

Appendix C - MS-DOS Version Specific

C.1 SuperView 100Report Software....C-1
C.2 Cyclic Report Requests....C-4
C.3 RFS File Transfer Utility Program....C-6
C.4 RFS Configuration Program....C-8
C.5 Error Messages Detected by the SuperView 100 Report Program....C-10
C.6 Error Messages from the RFS Server....C-11
C.7 Error Messages from the RFSFTU Program....C-12

Appendix D - Report Labels and Data Ranges

D.1 General....D-1
D.2 DATE Label and Data Range....D-2
D.3 TIME Label and Data Range....D-3
D.4 DSC Label and Data Range....D-4
D.5 TSC Label and Data Range....D-7
D.6 AI Label and Data Range....D-9
D.7 AO Label and Data Range....D-11
D.8 DI Label and Data Range....D-13
D.9 DO Label and Data Range....D-15
D.10 DAT Label and Data Range....D-17
D.11 MDAT Label and Data Range....D-18
D.12 TTD Label and Data Range....D-20
D.13 TTDO Label and Data Range....D-24
D.14 COM Label and Data Range....D-28
D.15 EVL Label and Data Range....D-31
D.16 EVLC Label and Data Range....D-34
D.17 TRIGG Label and Data Range....D-35

CONTENTS (continued)

Appendix E - SuperView 100 Windows Batch File Commands

E.1 ABSWAIT....E-1
E.2 RELWAIT....E-2
E.3 EXEC....E-3
E.4 SV100....E-4
E.5 GOTO....E-5
E.6 :<string>....E-6
E.7 GOSUB....E-7
E.8 RETURN....E-8
E.9 IF TRIGGIDENT=....E-9
E.10 IF ERROR....E-10
E.11 EXIT....E-11
E.12 **;***<comment>*....*E-12*

Appendix F - Syntax for DDE Subscriptions

F.1 General....F-1
F.2 AI Objects....F-2
F.3 AO Objects....F-4
F.4 DI Objects....F-6
F.5 DO Objects....F-8
F.6 DAT Objects....F-10
F.7 MDAT Objects....F-11
F.8 DSC Objects....F-13
F.9 TSC Objects....F-15

Appendix G - RFS

G.1 RFS Remote File Functions....G-1
  G.1.1 rfsdld....G-1
  G.1.2 rfsrep....G-3
  G.1.3 rfsuld....G-5
  G.1.4 rfsdel....G-6
  G.1.5 rfsrnm....G-7
  G.1.6 rfstra....G-8
  G.1.7 rfstcl....G-11
G.2 RFS Local File Primitives....G-13
  G.2.1 rfslop....G-13
  G.2.2 rfslcl....G-14
  G.2.3 rfsdel....G-15

CONTENTS (continued)

G.2.4 rfslrn G-16
G.2.5 rfslcr G-17
G.2.6 rfslgr G-18
G.2.7 rfslpr G-19
G.2.8 rfslsp G-20
G.2.9 rfslgf G-21
G.2.10 rfslcd G-22
G.3 RFS Error Log Function G-23
G.3.1 rfselr G-23
G.4 RFS DLL Functions (MS-Windows only) G-24
G.4.1 wrfscon G-24
G.4.2 wrfsdis G-25
G.4.3 wrfscnf G-25
G.5 Creating an RFS Application G-26
G.5.1 C-type Declarations G-26
G.5.2 Compile and Link G-27
G.5.3 C-code Example G-28

INDEX

Chapter 1 Introduction

1.1 General Information

This manual tells you how to install, configure and use SuperView 100, a software package designed for the Advant Station™ 100 series of IMS Stations (AS100 IMS). SuperView 100 logs process data from the ABB automation system and then generates reports based on that data. It runs on a personal computer (PC) under MS-DOS™/PC-DOS™ and, optionally, Microsoft® Windows™ to:

- Prepare spreadsheet reporting of process data
- Act as an interface for MS-Windows applications to data in the automation system
- Transmit data back to the automation system
- Provide file storage and transfer services

ABB automation system data is available to Lotus 1-2-3™, Microsoft Excel™ or file compatible spreadsheet software for worksheet calculations and reports through SuperView 100's spreadsheet report capabilities. Please note that the spreadsheet software is not included with SuperView 100.

SuperView 100's **Dynamic Data Exchange (DDE)** server interface can be used by other MS-Windows applications supporting DDE, such as Microsoft Excel, to access process data directly. Supported transmission of data back to the automation system enables you to send setpoints and download recipes for process control.

File storage and transfer services are available in SuperView 100 via the **Remote File System (RFS)**. You can transfer record-oriented sequential files in ASCII or binary form between AS100 IMS and other nodes in the ABB automation system.

SuperView 100 communicates with the rest of the automation system using ABB's **GCOM protocol**.

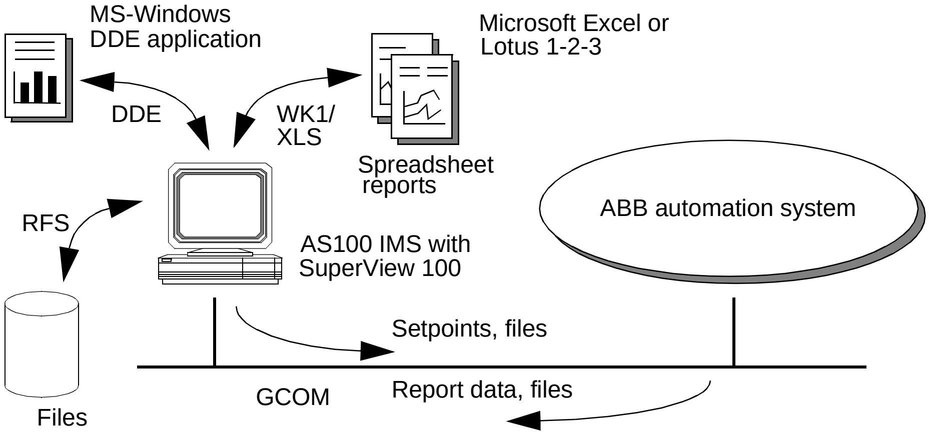


Figure 1-1. SuperView 100 Functions

1.2 Manual Organization

In this manual, you will find step-by-step instructions on how to install, configure and use SuperView 100 with spreadsheet software (for example Lotus 1-2-3 and Microsoft Excel) to generate reports based on data from the ABB automation system. You will also find descriptions of the various parts of the RFS and instructions explaining how to use the RFS to transfer files between AS100 IMS and other nodes in the automation system.

The typical users this manual is geared towards are:

- Engineers creating SuperView 100 report definition files, and
- Operators using SuperView 100 to generate reports and transfer files.

You should be familiar with application programming in ABB automation systems and Lotus 1-2-3, Microsoft Excel or file compatible spreadsheet software to fully utilize SuperView 100's report capabilities.

For the most part, this manual discusses SuperView 100 in the MS-Windows environment. Chapter 4, Operation, which describes the MS-Windows user interface, and part of Chapter 3, Configuration and Application Building, which outlines the DDE functions, are directed to MS-Windows users only. Information specific to MS-DOS users is found in Appendix C MS-DOS Version Specific.

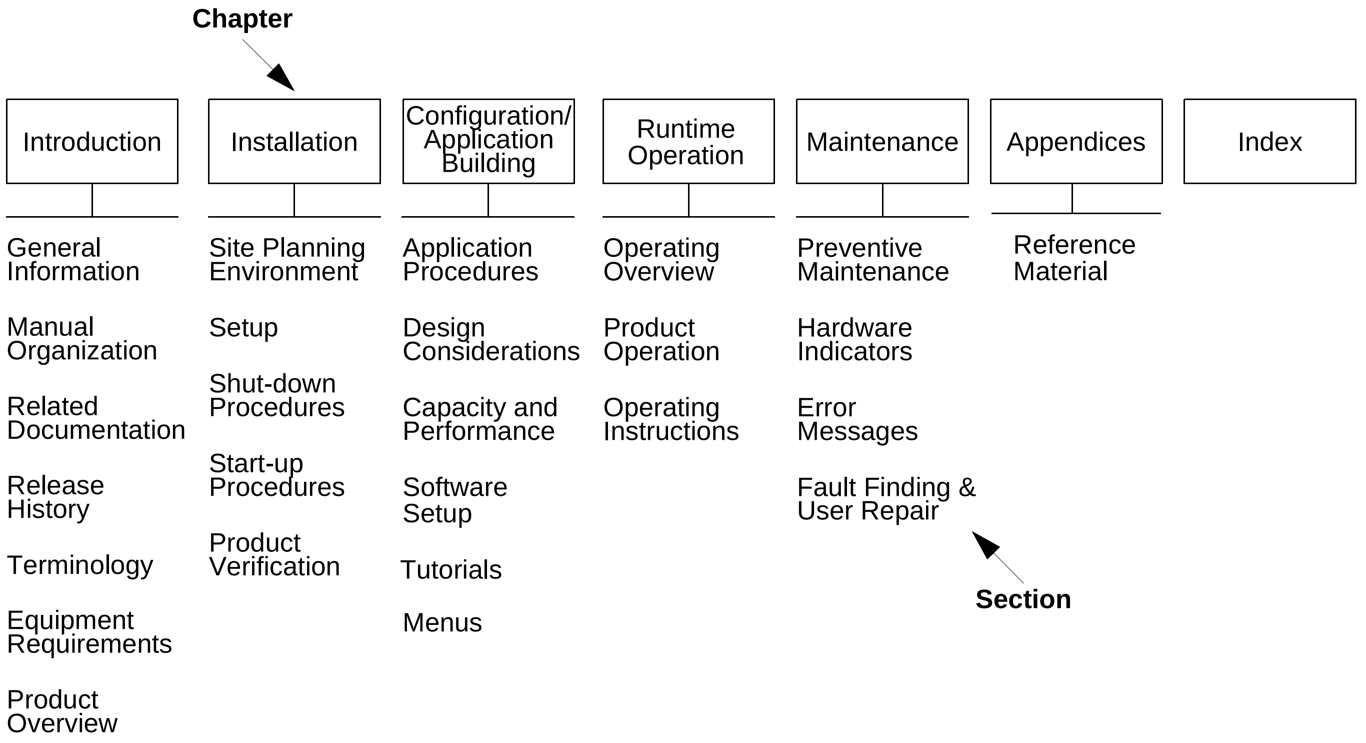


Figure 1-2. Manual Organization Diagram

Introduction

In Chapter 1, Introduction, you will find a brief description of SuperView 100’s functions, the hardware and software requirements, user requirements, related documentation, terminology, and this manual’s organization.

Installation

Chapter 2, Installation, tells you how to configure and install SuperView 100 in the PC. It also lists the environment requirements that you must fulfill before installation.

Configuration and Application Building

Chapter 3, Configuration and Application Building, describes how to create a report with spreadsheet files, DDE functions and SuperView 100 batch files that are used for report scheduling. You will also find a step-by-step tutorial and sections on report design considerations and capacity and performance.

Operation

Chapter 4, Operation, details the SuperView 100 user interface in the MS-Windows environment. You will find descriptions of all SuperView 100 Report and RFS Server software menus, dialog boxes and displays in this chapter.

Maintenance

Chapter 5, Maintenance, covers SuperView 100 error-handling.

Appendix A

Appendix A, Reference Material, describes report labels used in spreadsheet report definitions, the syntax for DDE links and the RFS programmer’s interface for creating your own user applications.

1.3 Related Documentation

Refer to the following documents for further information about how various data types are used in SuperView 100 and in the ABB automation system:

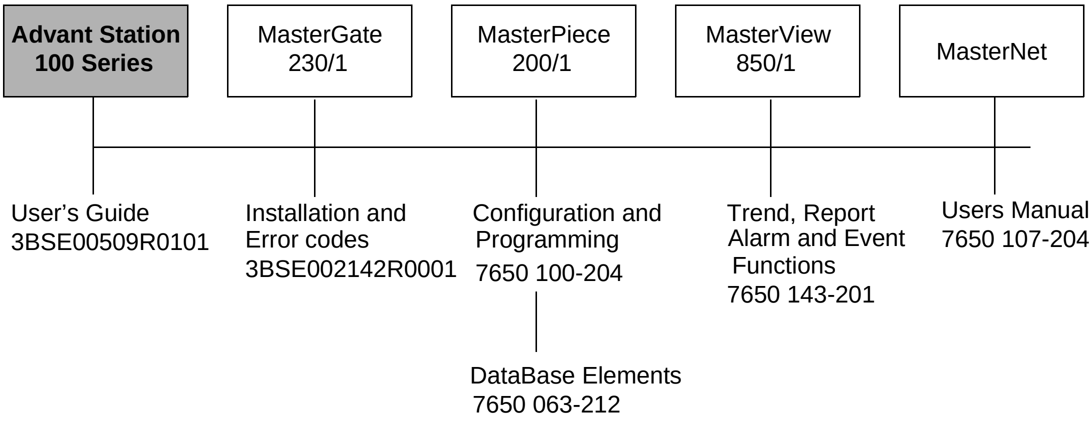

Advant Station 100 Series
MasterGate 230/1
MasterPiece 200/1
MasterView 850/1
MasterNet
User's Guide 3BSE00509R0101
Installation and Error codes 3BSE002142R0001
Configuration and Programming 7650 100-204
Trend, Report Alarm and Event Functions 7650 143-201
Users Manual 7650 107-204
DataBase Elements 7650 063-212

Figure 1-3. SuperView 100 and Related Documentation Tree

1.4 Release History

Major milestones in the SuperView 100 product release are:

Table 1-1. SuperView 100 Release History

Version	Description
1.0/0	Initial MS-DOS release
2.0/0	Extended with AI, AO, DI, DO, TTDO, EVL and TRIGG objects
3.0/0	MS-Windows release

1.5 Terminology

ABB automation system:
ABB automation system refers to one or all of the distributed control systems from ABB Process Automation, that is ABB Master, ABB MOD 300, ABB P4000 and ABB Procontrol p14.

Advant Station:
Advant Station is the general term of Advant Station series of operator, engineer and information management stations within ABB automation system.

DDE:
DDE is an abbreviation of Dynamic Data Exchange, a form of interprocess communication that uses shared memory to exchange data between applications in the MS-Windows environment.

DAT:
Dat is an object type in the databases of ABB automation system that is able to store different types of single-value application data. Also used in DSC data packets.

DSC:
DSC is an abbreviation of Data Set Communication, a function for transfer of application data packets between nodes in the ABB automation system.

MasterNet:
MasterNet is the name of the ABB Master Computer Network Communication System.

MasterGate (MG 230/1):
MasterGate (MG 230/1) is the name of the gateway stations for connection of ABB Master control networks to the plant network.

MasterPiece (MP 200/1):
MasterPiece (MP 200/1) is the name of the process stations in ABB Master.

MasterView (MV 800/1):
MasterView (MV 800/1) is the name of the operator stations in ABB Master released 1991 and older.

ModGate:
ModGate is the name of the gateway stations for connection of ABB MOD 300 networks to the plant network.

MultiDat (MDAT):
MultiDat (MDAT) is a number of consecutive DAT objects. MultiDat definitions are used efficiently to transmit suites of DAT instances in a single request.

RFS:
RFS is an abbreviation of Remote File System, a function for transfer of record-oriented sequential files in ASCII or binary form between nodes in the ABB automation system

TSC:
TSC is an abbreviation of Text Set Communication, a function for transfer of application texts between nodes in the ABB automation system.

TTD:
TTD is an abbreviation of Time tagged Data, a subsystem in ABB automation system for management of suites of current and historical data which is primarily used for display on trend curves.

1.6 Equipment Requirements

Basic equipment requirements for SuperView 100 are:

Basic Hardware

- PC with 80386 or higher CPU
- One 1.44 MByte 3.5" diskette drive
- A hard disk with at least 2 MByte of available disk space
- 2 MByte of RWM memory (recommended 4MByte)
- IEEE 802.3 communication board with NDIS interface support

Operating System

- MS-Windows 3.1 or later (for the MS-Windows version of SuperView 100) or
- MS-DOS/PC-DOS 3.0 or later (for the MS-DOS version of SuperView 100)

Application Software

- Spreadsheet software using the WK1 or the XLS file formats, such as Lotus 1-2-3, Microsoft Excel and Quattro Pro™
- MS-Windows application supporting DDE (for use of the DDE option provided by the MS-Windows version of SuperView 100)

1.7 Product Overview

The SuperView 100 software package consists of the following components:

- SuperView 100 Report software
- RFS Server software
- RFS file transfer library
- SuperView 100 Basic software
- GCOM driver

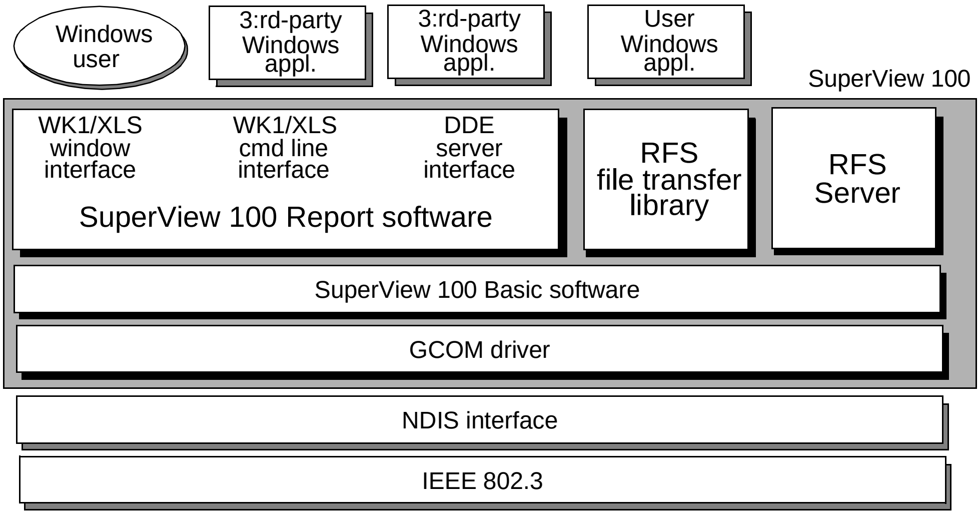


Figure 1-4. SuperView 100 and its MS-Windows Software Environment

SuperView 100 Report Software

SuperView 100 supports the Lotus 1-2-3 WK1 and the Excel XLS worksheet file formats. You have the full flexibility of spreadsheet software when you design and produce reports based on process data. SuperView 100 also has a DDE server interface to provide process data dynamically to other MS-Windows applications supporting DDE.

In addition, SuperView 100 Report software also offers a number of other functions:

- **Data transmission to the automation system.**

  With this feature, you can, for example, send setpoints and commands to the control system or notify process operators about developments as a result of calculations made by the spreadsheet program on the process data collected.

- **A trigger function to start report creation** when a predetermined event occurs in the process or process control system.

  You have the option to have reports created manually by process operators or automatically by process stations with this function.

- **Simple report sheduling using SuperView 100 batch files**.

  You can schedule reports by specifying report compilation at certain times or events, or cyclically within a given time interval using batch files. You can also launch other Windows or DOS applications from these files.

SuperView 100 Report software provides the following user interfaces:

- WK1/XLS Windows
- WK1/XLS Command Line
- DDE server

The WK1/XLS Windows interface offers convenient services for Windows users who want to create WK1/XLS reports interactively and schedule reports with a special type of batch file supported by SuperView 100.

The WK1/XLS Command Line interface can be used for interactive report compilation on command from other Windows applications, such as macros defined in spreadsheet software.

The DDE server interface can be conveniently used by other Windows programs supporting DDE to read process and calculated data from the rest of the automation system.

The following proprietary data types are supported,

Table 1-2. The Data Exchange Capabilities of SuperView 100

Data type	Into spreadsheet	DDE	Send data
AI / AO	yes	yes	no
DI / DO	yes	yes	no
DAT / MDAT	yes	yes	no
DSC / TSC	yes	yes	yes

Table 1-2. The Data Exchange Capabilities of SuperView 100

Data type	Into spreadsheet	DDE	Send data
TTD	yes	no	no
COM	yes	no	no
EVL	yes	no	no

where:

- **AI / AO** are analog process inputs and outputs (or calculated values).
- **DI / DO** are digital process inputs and outputs (or calculated values).
- **DAT /MDAT** are general process, or calculated, data. MDAT is a mode of subscribing to consecutive DAT records in suites, as opposed to one by one, to improve communication efficiency.
- **DSC / TSC** are general data and text packets exchanged between application programs in the stations of the ABB automation system.
- **TTD** are files of time-tagged, historical process (or calculated) data.
- **COM** are records of node and network-communication status information.
- **EVL** are files of time-tagged, historical, event and alarm messages.

RFS Server

The RFS Server transfers any record-oriented sequential file (ASCII or binary) from AS100 IMS to a remote node (for example MasterBatch) or from a remote node to AS100 IMS at the initiative of the remote node. Files are transferred on a client-server basis.

RFS File Transfer Library

The RFS file transfer library consists of high-level functions which can be called up by user applications to:

- Transfer any record-oriented sequential file (ASCII or binary) from AS100 IMS to a remote node and from a remote node to AS100 IMS.
- Translate logical report names to physical report names in MOD 300™.
- Carry out TCL catalog functions on sequences and recipes in MOD 300.

These library functions free the user from the details of message creation, transmission, receipt and communication with the remote system. See Section G.1 RFS Remote File Functions.

SuperView 100 Basic Software

SuperView 100 Basic software provides the fundamental functions for communicating with the ABB automation system. It is used by SuperView 100 Report software, the RFS Server and the RFS file transfer library. SuperView 100 Basic software, however, cannot be used simultaneously by two applications, meaning that SuperView 100 Report software, the RFS Server or a user RFS application cannot coexist.

The GCOM Driver

The GCOM driver is a DOS device driver implementing the GCOM protocol. The NDIS interface for the GCOM driver provides you with the ability to use any IEEE 802.3 communication board with NDIS support. Other communication protocol drivers (for example TCP/IP) with NDIS interfaces can share the same IEEE 802.3 board.

Chapter 2 Installation

2.1 Site-Planning Environment

Before installing SuperView 100, make sure that your PC meets the following requirements:

- MS-DOS 3.0 (or later) and MS-Windows 3.1 (or later) operating systems are installed.
- The IEEE 802.3 communication board and the corresponding NDIS driver are installed (see Section 2.2.1 Hardware).

In configurations using MP 200/1, the MP 200/1 must be equipped with options for MV 800/1 functionality in the following cases:

- If the MP 200/1 is equipped with a GCOM board for connection to SuperView 100.
- If data other than DSC and TSC is to be collected from the MP 200/1.

2.1.1 SuperView 100Configuration

It is important that you know the appropriate GCOM configuration to use. You must assign the following variables during the SuperView 100 installation:

- The SuperView 100 address (**NETW**, **NODE**).
- The GCOM transit node (**TRNODE**).
- The ABB Master networks to communicate with (**CTRLNETW**).
- The SuperView 100 identification (**SV100_ID**).

These variables are permanently defined in the DOS environment from the **AUTOEXEC.BAT** file, which is automatically updated during the installation.

SuperView 100Address

The network and node address of the SuperView 100 node in the GCOM subnet is necessary for communication with the ABB automation system and must be defined. This node address follows the rules of the **ABB MasterNet**. It is specified by the parameters NETW and NODE, which are limited to:

- NETW=11-99 (control network connection) or 110-119 (plant network connection)
- NODE= 1-99

See Appendix A Network and Node Configuration Rules, for the rules for the GCOM network and node address. Required communication address parameters are described here along with examples of various network configurations.

GCOM Transit Nodes

Specify the transit node only if SuperView 100 is to communicate with an ABB Master® control network. The transit node must be equipped with a DSCS150 communication board and act as a gateway to the other nodes in the control network. Only ABB MasterPiece 200/1 or MasterGate 230/1 nodes in the GCOM subnet can be transit nodes.

To facilitate communication with other nodes in an ABB Master control network, the transit node must be specified by the parameter TRNODE and is limited to:

TRNODE=1..99, 127

If communication with more than one transit node is desired, specify the numbers of these transit nodes with commas or colons as delimiters as follows:

TRNODE=1,2,3 (or **1:2:3)**

If SuperView 100 is to communicate only with nodes within the same GCOM network, you do not have to specify the TRNODE parameter.

ABB Master Networks

The ABB Master control networks with which SuperView 100 is required to communicate are defined by their control network numbers. A control network number is defined as the tenth of the network address. For example, if your GCOM network number is 15, the control network number is 1, for network number 24 it is 2 and so on.

Specify control network numbers to improve communication and minimize communication overhead. Specify control networks by the parameter CTRLNETW, which is limited to:

CTRLNETW=1..9

If communication with more than one control network is desired, the numbers of these networks must be specified with commas or colons as delimiters as follows:

CTRLNETW=1,2,3 (or **1:2:3**)

If SuperView 100 is to communicate only with nodes within the same GCOM network, the CTRLNETW parameter does not have to be specified.

SuperView 100 Identification

SuperView 100 requires a unique identification to distinguish it from other SuperView systems. **Event log data requests (EVL)** use this identifier to keep track of which process events have been sent and the SuperView system to which they have been sent. This identification is specified by the parameter SV100_ID and is limited to:

SV100_ID=1..8

If the SuperView 100 identification is not specified, it will be set up to 5, the default value.

2.2 Setup

2.2.1 Hardware

SuperView 100 does not come with any hardware. Some useful hints follow, however, to help you install the IEEE 802.3 board to the GCOM network.

- You may use the 3Com505 board included with SuperView 100 Releases 1.0 and 2.0. If, however, you will be installing MS-Windows software, you must replace your old GCOM and 3Com505 board drivers with the corresponding NDIS drivers (GCOMDRV.SYS and ELNKPL.DOS).
- Install the IEEE 802.3 communication board according to the instructions given in the IEEE 802.3 communication board’s manual. Make sure that there is no conflict with other devices regarding the I/O address, interrupt level (IRQ) and RWM base address. Also check that the jumper position for the use of thin (10Base2) and thick (10Base5) Ethernet cable, respectively, is correct.
- If you use an external transceiver (thick Ethernet), make sure that the jumper for heartbeat control is set to *on*.

2.2.2 Software

It is extremely important that you **know the software setup requirements for your system configuration before installing SuperView 100**. See Section 2.1.1 SuperView 100Configuration, to determine the software configurations and setups you will require before installing the software.

SuperView 100 Distribution Files

The SuperView 100 installation diskette contains the following files.

Files for both **DOS** and **MS-Windows** versions:

- **README.TXT**

  Contains the latest information on the product, the installation procedure and useful tips.
- **GCOMDRV.SYS**

  Contains the SuperView 100 GCOM device driver with an NDIS interface.
- **ELNKPL.DOS**

  Contains the NDIS driver for the 3Com505 board.
- **PROTMAN.DOS and PROTMAN.EXE**

  Contains the protocol manager device used by NDIS.
- **NETBIND.COM**

  Makes the protocol bindings defined in the file PROTOCOL.INI.

- **PROTOCOL.INI**

  Example file with the GCOM NDIS driver.

Files for **MS-Windows version** only:

- **INSTALL.EXE**

  Contains the SuperView 100 installation program for the MS-Windows environment. Copies all SuperView 100 files to the hard disk and updates the AUTOEXEC.BAT and CONFIG.SYS files. Also creates a SV100 file group, with the SV100 Report and RFS Server software, in MS-Windows.

- **SV100WIN.EX**E

  Contains the SuperView 100 Report program.

- **RFSSRVWN.EXE**

  Contains the SuperView 100 RFS Server program.

- **SV100DLL.DLL**

  Contains the SuperView 100 Basic software.

- **RFSWIN.H**

  Contains the RFS C-type declarations.

- **RFSAPI.DLL**

  Contains the RFS file transfer library.

Files for the **MS-DOS version** only:

- **SV100INS.BAT**

  Contains the SuperView 100 installation program for the MS-DOS environment. Copies all SuperView 100 files to the hard disk and updates the AUTOEXEC.BAT and CONFIG.SYS files.

- **SV100SET.BAT**

  Contains the SuperView 100 configuration setup program. Complements the installation program and changes the setup information in AUTOEXEC.BAT.

- **SV100.EXE**

  Contains the SuperView 100 Report program.

- **RFSSRV.EXE**

  Contains the SuperView 100 RFS Server program.

- **RFSFTU.EXE**

  Contains the SuperView 100 RFS File Transfer Utility program.

- **RFSCNF.EXE**

  Contains the SuperView 100 RFS configuration program.

- **RFS.H**

  Contains RFS C-type declarations.

- **RFS.LIB**

  Contains the RFS file transfer library.

- **ABSWAIT.COM, RELWAIT.COM**

  Contains SuperView 100 DOS batch file extenders for cyclic report requests through DOS batch files.

Installing the Software

Read the README.TXT file on the distribution diskette for any additional information regarding distribution before installing the software.

To set up SuperView 100 for the **MS-Windows environment**:

1. Start MS-Windows.
2. Insert the SuperView 100 distribution diskette in the drive.
3. Choose **Run** from the Windows **File** menu.
4. Enter the appropriate drive letter, followed by **:\install**, and press **Enter**.

   For example, the command line might look like this,

   ```
   a:\install
   ```

5. Follow the instructions on the screen.

To set up the SuperView 100 for the **MS-DOS environment**:

1. Insert the SuperView 100 distribution diskette in the drive.
2. Enter the appropriate drive letter, followed by a colon (:), at the DOS prompt and press **Enter**.

   For example, the command line might look like this,

   ```
   C> a:
   ```

3. Enter **sv100ins** and press **Enter**.
4. The parameters needed for installation are displayed and described. Proceed with the installation by entering the required parameter values. For example,

   ```
   A>sv100ins parameter parameter parameter ...
   ```

5. Follow the instructions given in the installation program.

The installation programs create a subdirectory named SV100 (if not already existing) on the selected hard disk drive and copies all SuperView 100 files there. An RFS subdirectory which will be used by the RFS Server is also created. Finally, the installation program updates the DOS system files AUTOEXEC.BAT and CONFIG.SYS.

When you install SuperView 100 software, it must be configured for the communication addresses to be used. You are guided through all the required settings by the installation program.

NOTE

After installation of the SuperView 100 software, the PROTOCOL.INI file must be manually updated with binding between the GCOM device and the IEEE 802.3 board driver. If the SuperView 100 installation created a new PROTOCOL.INI file, it must manually be updated with the IEEE 802.3 board driver. You must also check that the IEEE 802.3 board driver is loaded, after the NDIS protocol manager driver (PROTMAN), in the CONFIG.SYS file. Please refer to your IEEE 802.3 board documentation.

Changes to the AUTOEXEC.BAT File

For SuperView 100 to communicate with an ABB Master control network, you must configure the network and node addresses of SuperView 100 and of the transit station to be used. These parameters are permanently defined by a **SET** command in the AUTOEXEC.BAT file. The SuperView 100 installation program will automatically update these parameters.

Define the following environment variables to establish communication with the ABB automation system:

NETW=<*The GCOM network address of the PC station*>

NODE=<*The GCOM node address of the PC station*>

Define the following environment variable to establish communication between SuperView 100 and an ABB Master control network:

TRNODE=<*Transit node address in the ABB Master control network*>

Define the following environment variable for the best possible communication performance in an ABB Master control network:

CTRLNETW=<*Control network number(s)*>

Set the default directory where SuperView 100 will read all configuration files with the following environment variable:

SVFILES= <*path*>

Define the following environment variable to use SuperView 100’s event log function (EVL):

SV100_ID= <*an identity number between 1 and 8*>

An environment variable for the RFS is required. Define the RFS root directory, used by the RFS server for remote file handling, as follows:

SV_RFSDIR=<*drive letter*>:\RFS

Define the default configuration file, used by the RFS configuration program, as follows:

RFS_CONF=<*drive letter*>:\RFS\RFSCNFE0.DAT

If NDIS drivers are used the NETBIND program must be called to make the protocol bindings:

<*path*>\NETBIND

Set up the PATH to the SV100-created directory as follows:

PATH=%PATH%;<*path*>

The AUTOEXEC.BAT file might look as follows after a SuperView 100 installation:

```
@ECHO OFF
SET NETW=15
SET NODE=60
SET TRNODE=21
SET CTRLNETW=1,2
SET SVFILES=C:\SV100
SET SV100_ID=5
SET SV_RFSDIR=C:\RFS
SET RFS_CONF=C:\RFS\RFSCNFE0.DAT
C:\SV100\NETBIND
.
.
PATH=%PATH%;C:\SV100
```

Figure 2-1. Example of the AUTOEXEC.BAT File After Installation of SuperView 100

Changes to the CONFIG.SYS File

Update the CONFIG.SYS file with the GCOM driver used by SuperView 100:

DEVICE=C:\SV100\GCOMDRV

After a SuperView 100 installation, the CONFIG.SYS file might look as follows:

```
DEVICE=C:\SV100\PROTMAN.DOS I:C:\SV100
DEVICE=C:\SV100\GCOMDRV.SYS
DEVICE=C:\SV100\ELNKPL.DOS
.
.
```

Figure 2-2. Example of the CONFIG.SYS File After Installation of SuperView 100

NOTE

Install the GCOM and IEEE 802.3 board driver after the Protocol Manager driver (PROTMAN.DOS) in the CONFIG.SYS file. The I:<*path*> parameter after PROTMAN.DOS refers to the directory where the PROTOCOL.INI file is found.

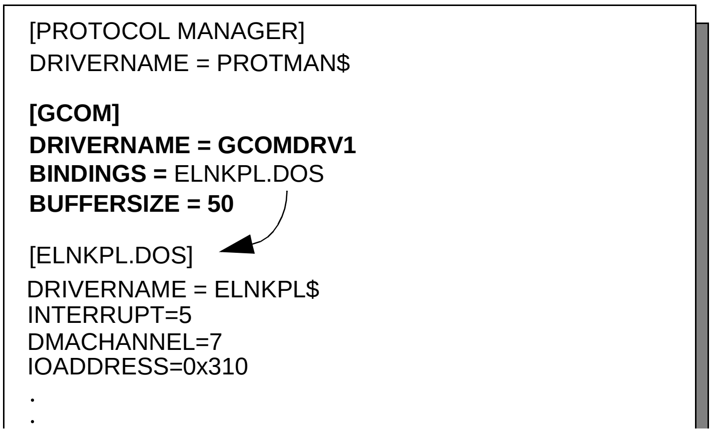
Changes to the PROTOCOL.INI File

The Protocol Manager uses the PROTOCOL.INI file to bind the NDIS drivers. Update this file with the GCOM driver and its bindings to the IEEE 802.3 driver. Set up a variable (BUFFERSIZE) for the queue size in the device for transmitted and received messages. If the BUFFERSIZE specification is omitted, the default queue size is 10 kByte. The actual size of the message queue can never exceed 50 KByte.

Update the PROTOCOL.INI file with the following lines:

- **[GCOM]** defines the start for the GCOM driver description.
- **DRIVERNAME=GCOMDRV1** specifies the name of the GCOM driver, which is GCOMDRV1.
- **BINDINGS=*board driver*** describes the binding to the IEEE 802.3 driver.
- **BUFFERSIZE= 50** sets up the message queue to 50 KByte.

The PROTOCOL.INI file with a GCOM driver might look as follows:



```
[PROTOCOL MANAGER]
DRIVERNAME = PROTMAN$

[GCOM]
DRIVERNAME = GCOMDRV1
BINDINGS = ELNKPL.DOS
BUFFERSIZE = 50

[ELNKPL.DOS]
DRIVERNAME = ELNKPL$
INTERRUPT=5
DMACHANNEL=7
IOADDRESS=0x310
.
.
```

Figure 2-3. Example of PROTOCOL.INI File After Installation of SuperView 100

NOTE

The PROTOCOL.INI file is **NOT** automatically updated by the SuperView 100 installation program. It must be modified manually.

2.3 Shutdown Procedures

Shut down SuperView 100’s programs as follows.

For the **MS-Windows** version:

- Choose **Exit** from the **File** menu or double-click in the upper left-hand corner of the screen to stop the SuperView 100 Report program.
- Select the icon and then choose **Close** from the **System** menu to stop the RFS server program.

For the **DOS** version:

- Press **<CTRL>**+**<BREAK>** at the DOS command line.

2.4 Start-Up Procedures

Start up SuperView 100’s programs as follows.

For the **MS-Windows** version:

1. Double click on the SV100 program group.
2. Double click on the **Program** icon.

For the **DOS** version:

Enter the following commands at the DOS command line.

1. **SV100 <*parameters..*>** (to start the SuperView 100 Report program)
2. **RFSSRV** (to start the RFS Server)
3. **RFSFTU** (to start the RFS File Transfer Utility program)
4. **RFSCNF** (to start the RFS Configuration program)

2.5 Product Verification

To verify installation and that SuperView 100 has contact with the ABB automation system, do the following.

For the **MS-Windows** version:

1. Double-click on the SuperView 100 Report icon to start the SuperView 100 Report program.
2. Choose **Comm Stat** from the **Option** menu.
3. Verify that at least one TRNODE is listed in the communication status dialog box.
4. Choose **About** from the **Help** menu.
5. Verify that your SuperView 100 version and registered licence is correct.

For the **DOS** version:

1. Enter **C:> SV100 /STATUS** at the DOS command line to start the SuperView 100 Report program.
2. Verify that at least one TRNODE is listed on the screen.
3. Verify that your SuperView 100 version and registered licence is correct.

Chapter 3 Configuration/Application Building

3.1 Application Procedures

In conjunction with Microsoft Excel, Lotus 1-2-3 or another file compatible spreadsheet software, SuperView 100 is a powerful tool for generating reports based on data from the ABB automation system.

The following SuperView 100 functions will be described in this chapter:

- Report production with spreadsheet software
- Batch files for report sheduling
- How to use DDE for access to process data
- The RFS Server
- The RFS file transfer library

Examples in this chapter use Microsoft Excel spreadsheet software. Descriptions, however, are applicable to any other spreadsheet package you choose.

3.1.1 Report Production with Spreadsheet Software

Follow these three steps to create reports based on ABB automation system data using SuperView 100:

1. Define the report.
    a. Create a report definition worksheet using a spreadsheet.
    b. Save the definition as a Lotus WK1 (or Microsoft Excel XLS) format worksheet file.
2. Request data for the report.
    a. Submit the report definition worksheet file to SuperView 100 Report software to create a report data worksheet file.
3. Generate a report.
    a. Retrieve the worksheet file with appropriate formatting expressions, formula and macro definitions. This file could be the report definition worksheet, but could also be any predefined worksheet or macro file.
    b. Combine this file(s) with the report data worksheet in the spreadsheet.
    c. Create the desired report using spreadsheet functions.

These three steps are explained in greater detail, along with examples, in the following sections.

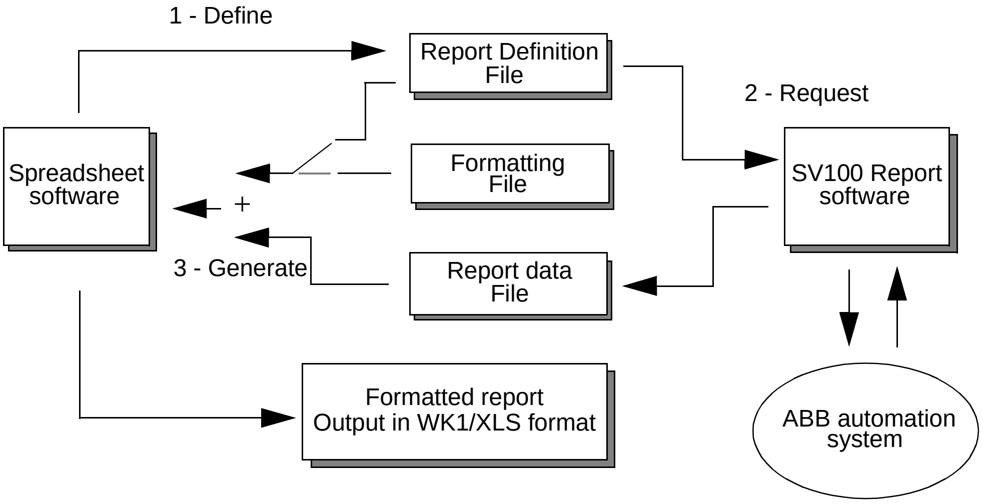


Figure 3-1. The SuperView 100 Report Production Cycle

3.1.1.1 Defining a Report

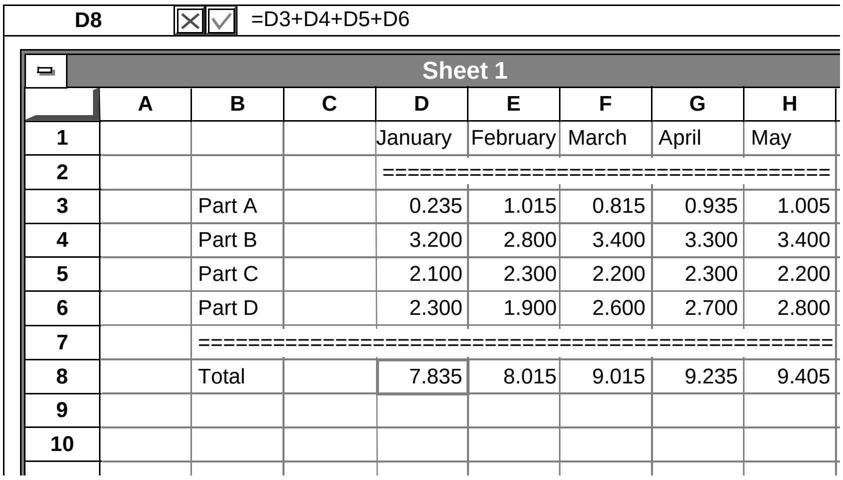
Reports are defined using spreadsheet software such as Lotus 1-2-3 or Microsoft Excel. Once a report is defined, a worksheet file is created. This file is called a **report definition file**. It contains the report definition and labels, called **report labels**, which define references to the data required for the report.

Report definition files also contain data that is to be sent rather than collected. They may contain legal file format records such as formulas needed for calculations, etc. Data dispatch can be defined as formulas referring to other report definition data.

Spreadsheet Software

Spreadsheet software is a powerful tool for worksheet calculations and reporting. You can use it to flexibly organize, process and analyze data from the ABB automation system. You can also create graphs based on this data in a variety of formats. In addition, data can be exported to other types of software that uses other file formats.

The basic data structure for storing and organizing information in spreadsheet software is the worksheet. The worksheet consists of **columns**, identified by letters, and **rows**, identified by numbers. A **cell** is the intersection of a column and a row. The cell is the smallest unit of the worksheet where data can be entered and stored. A column letter followed by a row number, for example C12, defines the location of a cell—or the **cell address**.



D8 ☒ ☑ =D3+D4+D5+D6

Sheet 1

	A	B	C	D	E	F	G	H
1				January	February	March	April	May
2				====================================				
3		Part A		0.235	1.015	0.815	0.935	1.005
4		Part B		3.200	2.800	3.400	3.300	3.400
5		Part C		2.100	2.300	2.200	2.300	2.200
6		Part D		2.300	1.900	2.600	2.700	2.800
7		===						
8		Total		7.835	8.015	9.015	9.235	9.405
9								
10								

Figure 3-2. A Typical Worksheet Screen

The contents of an individual cell can be a label, a number or a formula. Labels are normally used to describe data in other cells. Numbers represent the data. Formulas are used to obtain new data based on data in other cells and can process data in many different ways.

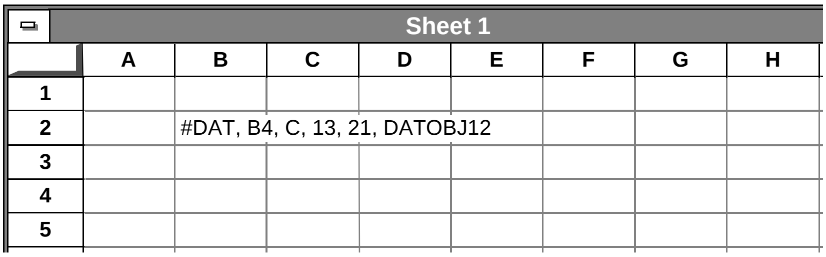
In Figure 3-2, you will find the following types of cells:

- Cells B3, B4, B5, B6, B7 and B8 are label cells.
- Cells D3, D4, D5 and D6 are number cells.
- Cells D8, E8, F8, G8 and H8 are formula cells.

Spreadsheet software also has a macro facility that you can use to automate repetitive or complicated command sequences, which increases productivity and decreases the risk of errors.

Report Labels

Report definition worksheets contain label cells which reference the ABB automation system objects whose data is used in reports. These label cells are referred to as **report labels**. They are used by SuperView 100 Report software to generate requests for ABB automation system data.



Sheet 1

	A	B	C	D	E	F	G	H
1								
2		#DAT, B4, C, 13, 21, DATOBJ12						
3								
4								
5								

Figure 3-3. Report Label Definition

Each report label contains a number of parameters, separated by commas or blanks, starting with a report label tag **#**, followed by an object-type identifier, for example **DAT** for a DAT object request. The report label identifier, then, for a DAT object request is **#DAT**.

Report labels define the source objects for data as well as how that data will be organized. For this purpose, report labels contain a cell reference that defines where the first unit of data is to be stored. Report labels also contain information about how various pieces of data are to be organized, that is, by row (all data items on the same row, **R**) or by column (all units of data in the same column, **C**). For example, begin a request to put DAT object data in cell E12 (column E, row 12), organized by row, as follows:

#DAT,E12,R ...

Since this data is organized by row, the cells containing the result in the result worksheet will be labelled E12, F12, etc. The report label identifier and the address information are followed by parameters defining the source object for the data item, that is, the network and node numbers and the name of the object in the ABB automation system, all separated by commas or blanks. The parameters defining the source object and the data structure are unique to the type of object (the report label identifier).

The label definition in Figure 3-3 can then be translated to the following: Request for a DAT object where the data is put in cell B4 with columnwise organization. The DAT object is named DATOBJ12 and exist in network 13, node 21.

See Appendix D Report Labels and Data Ranges, for a detailed description of report label layouts and their parameters.

Entering Report Labels

Enter report labels into report definition worksheets in exactly the same way as any other label in the spreadsheet. Since report labels are normally longer than the default column width of the worksheet, they usually overrun the column boundaries. A report label must always be completely defined within a single cell, not split into several cells.

NOTE

> Report labels are always defined as **text strings**. In some spreadsheet software, they must start with a **’** prefix to be left-adjusted.

Saving Report Definitions

Save a report definition worksheet as a Lotus WK1 or Excel XLS file format worksheet file, on disk, to make it retrievable for subsequent SuperView 100 report requests. A report definition worksheet is saved like any other worksheet.

NOTE

> If Lotus 1-2-3 is used, the cell pointer should be in its home position (cell A1) before you save a report definition. This is to ensure correct cell references between files, when files are combined later.

3.1.1.2 Requesting Report Data

You can request compilation and output of a report after a report definition worksheet file has been created.When you request a report, the report definition file is submitted to SuperView 100 Report software, which interprets it and generates requests for data to the ABB automation system.

When the data is received from the ABB automation system, it is interpreted by the SuperView 100 Report software and put into a new worksheet file called the **report data file**.

Report Data Files

Report data files created by SuperView 100 Report software contain all data, in data cells, received in response to a single report request. Report data files are produced in Lotus 1-2-3 WK1 or Excel XLS file format, depending on the file format of the report definition file. The data, obtained as a result of each report label, is stored in a cell range where the upper left-hand cell is defined in the label, and expanding downwards or sideways according to the definition in the report label.

A report data file is given the file name of the report definition used to create it, plus a name extension, **.WKD** (or **.XLD**). Exceptions to this are:

- Report data files for reports with the **EVLC** label, which are given the same file name as the report definition file, but assigned the extensions .001, .002, .003, etc.
- Report data files for reports with the **TRIGG** label, where the name is given in the label and the file extensions are .001, .002, .003, etc.
- Calls to SuperView 100 Report software using the **/Outfile=<*file*>** qualifier. The /Outfile=<file> qualifier is described more fully in Section 4.2.1.2 The Command Line Interface.

3.1.1.3 Creating Reports

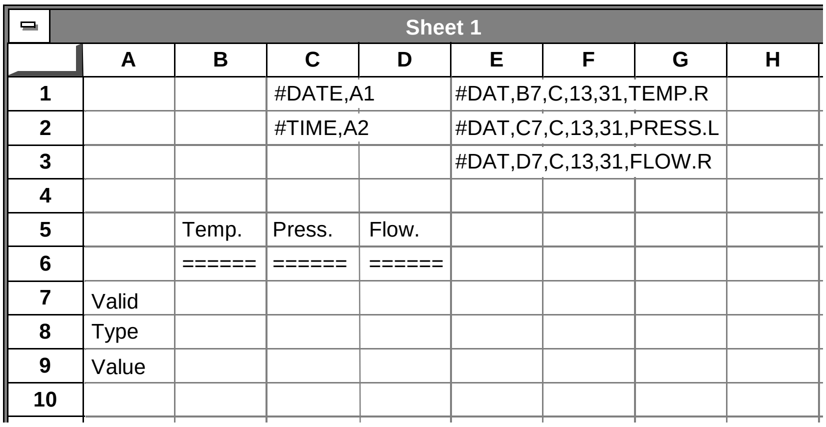
Once you have requested data for a report and created a report data file, the final report can be compiled. The appearance and contents of the report are determined by the formulas, macros, labels and graphs in the report definition worksheet. You can include headings, labels, tables, etc., in report definitions to format reports for output. Formulas included in the report definitions calculate the desired values.

Spreadsheet software defines reports, retrieves report definition worksheets and combines them with the associated report data worksheets, and processes data to produce report output in a suitable format.

Retrieving Report Definitions

To generate a report, in most cases, you must retrieve the report definition before retrieving the report data. This is necessary if the report definition contains formulas, format labels or lines referring to the report data. Another worksheet or macro file might also be used for the report formatting.

Once you have retrieved the report definition file, the worksheet screen might look like the example in Figure 3-4 below.



	A	B	C	D	E	F	G	H
1			#DATE,A1		#DAT,B7,C,13,31,TEMP.R			
2			#TIME,A2		#DAT,C7,C,13,31,PRESS.L			
3					#DAT,D7,C,13,31,FLOW.R			
4								
5		Temp.	Press.	Flow.				
6		======	======	======				
7	Valid							
8	Type							
9	Value							
10								

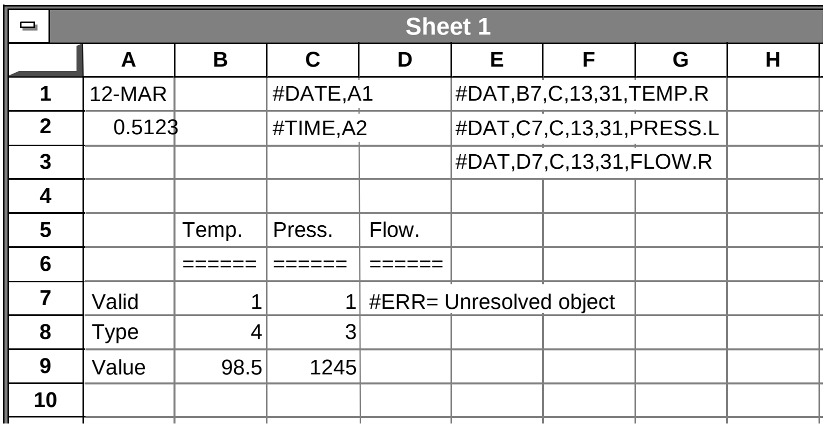
Figure 3-4. A Report Definition Worksheet

Incorporating Report Data

The next step is to incorporate all data from the report data file into the worksheet used for the final report (in this case the report definition file). You can accomplish this in a number of different ways. For example, you can:

- Open the file and activate a macro.
- Append the file to a formatting file.
- Use the clipboard.

Data can be incorporated other ways as well. After you have retrieved all the report data to the report definition file, the worksheet screen might look like the example in Figure 3-5 below.



Sheet 1

	A	B	C	D	E	F	G	H
1	12-MAR		#DATE,A1		#DAT,B7,C,13,31,TEMP.R			
2	0.5123		#TIME,A2		#DAT,C7,C,13,31,PRESS.L			
3					#DAT,D7,C,13,31,FLOW.R			
4								
5		Temp.	Press.	Flow.				
6		======	======	======				
7	Valid	1	1	#ERR= Unresolved object				
8	Type	4	3					
9	Value	98.5	1245					
10								

Figure 3-5. A Report Definition Worksheet with Retrieved Report Data

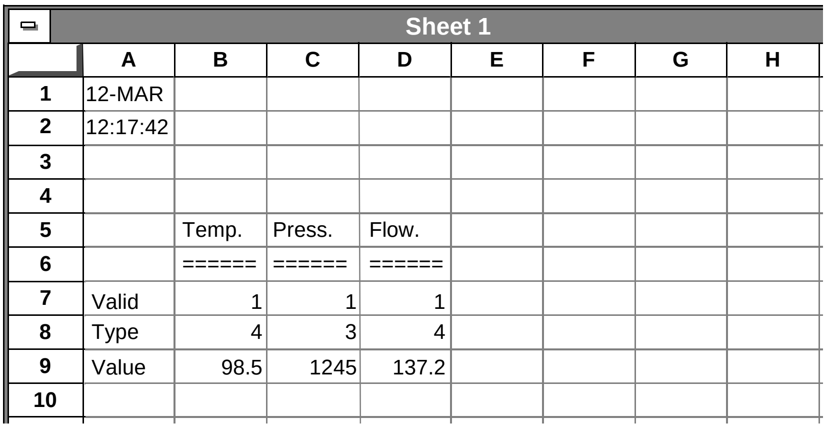
Any errors in report labels or lack of data are reported through error labels in the first data cell of the data range referenced. Error labels identify errors. An example of an error label is shown in cell D7 (#ERR= Unresolved object) of the combined worksheet in Figure 3-5 above. In this example, the wrong name for an object has been specified.

Error labels are described in detail in Section 5.3.1 Error Labels.

Generating Report Output

When the final report is generated, you have access to the full power and flexibility of spreadsheet software to create reports for print out or disk storage. Before creating the report output file, you may need to format some data. One example of this is stamping the date and time in reports, which is done to the internal Excel/Lotus format by default.

You may also need to erase unnecessary parts of the worksheet, such as those containing formulas, definitions, etc., before outputting the report. Once you have completed formatting the report for output, the worksheet screen might look like Figure 3-6 below.



	A	B	C	D	E	F	G	H
1	12-MAR							
2	12:17:42							
3								
4								
5		Temp.	Press.	Flow.				
6		======	======	======				
7	Valid	1	1	1				
8	Type	4	3	4				
9	Value	98.5	1245	137.2				
10								

Figure 3-6. A Processed Report Definition Worksheet

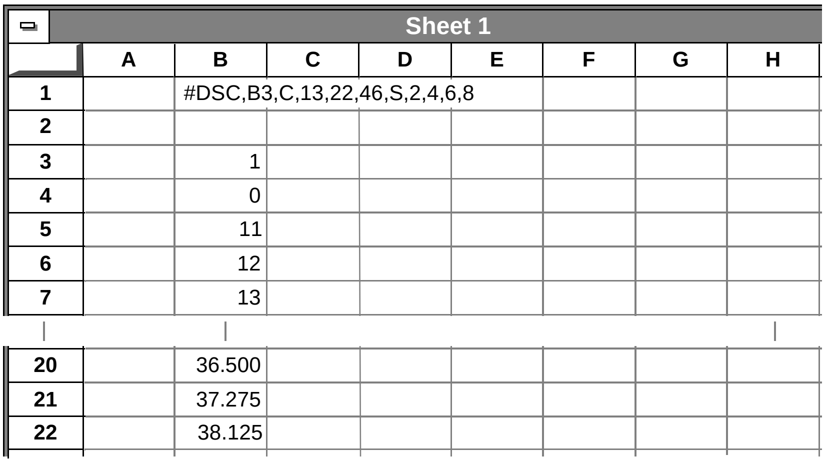
Formatting reports before output is a typical macro application that increases productivity.

3.1.1.4 Sending Data to the ABB Automation System

In addition to collecting data from the ABB automation system, you can also send data to the ABB automation system from SuperView 100. Data transmission is supported by the DSC and TSC label’s **send** function, which permits data set and text set packages to be sent to the ABB automation system.

Transmission of data is defined by normal DSC or TSC report labels in report definition files. A report label has a parameter identifying it as a sending report label. The parameter defines whether the report label is sending (**S**) or receiving data (**R**).

For example, to send DS data to a receiving DS in node 22, network 13, with the DS identity 46, starting from cell B3 by column, consisting of two Booleans, four integers, six long integers and eight reals, the label should be defined as in Figure 3-7.



Sheet 1

	A	B	C	D	E	F	G	H
1		#DSC,B3,C,13,22,46,S,2,4,6,8						
2								
3		1						
4		0						
5		11						
6		12						
7		13						
20		36.500						
21		37.275						
22		38.125						

Figure 3-7. A DSC Report Label for Transmission of Data

The **S** parameter defines the report label as *sending*. The data to be sent must be found in the report definition and in the data range referred to from the report label. Two sending DSC (or TSC) labels CAN NOT, however, share the same data cell.

In this example, the data range consists of the following:

- Cells B3 and B4 must contain two Boolean values.
- Cells B5 to B8 must contain four integer values.
- Cells B9 to B14 must contain six long integer values.
- Cells B15 to B22 must contain eight real values.

See Appendix D Report Labels and Data Ranges, for a detailed description of the layout of report labels for sending DSC and TSC labels, their parameters and the layout of the data range.

3.1.2 Report Scheduling with SuperView 100 Batch Files

You can create simple report schedules using SuperView 100 batch files. Reports can be schedules for production at specific times or within certain time intervals. You can also run other MS-Windows or DOS applications (restricted to .EXE files only) from these batch files.

When the SuperView 100 Report software is executing a batch file, you can always stop and restart it. Batch file execution is cyclical, which means that batch execution will automatically start from the first line after the last line is executed. To avoid this, you must use the **EXIT** batch command (see Appendix E SuperView 100 Windows Batch File Commands).

Use the MS-Windows NotePad, or any other text editor, to create a SuperView 100 batch file. Name it with the extension **.SV2** so it will be listed automatically in the Batch File Selection dialog box for batch file selection (refer toSection 4.2.1.1 The Windows Interface).

Batch File Commands

You can use the following commands in a SuperView 100 batch file:

- **ABSWAIT** <*hh:mm*>
- **RELWAIT** <*hh:mm*>
- **EXEC** <*application*>.EXE
- **SV100** <*file*> </*qualifiers...*>
- **GOTO** <*string*>
- ***<string>:***
- **GOSUB** <*string*>
- **RETURN**
- **IF TRIGGIDENT=**<*integer value*>
- **IF ERROR**
- **EXIT**
- ***;<comment>***

For more information about the different batch file commands please refer to Appendix E SuperView 100 Windows Batch File Commands.

3.1.3 Using Dynamic Data Exchange (DDE)

DDE actively exchanges data between MS-Windows applications. Using DDE and SuperView 100 Report software, you can:

- Retrieve data from the ABB automation system.
- Send data to the ABB automation system.
- Carry out commands in the SuperView 100 Report software.

Request for Process Data

The DDE interface can be used by other MS-Windows applications, with DDE support, to collect data from the ABB automation system. SuperView 100 Report software acts as the DDE server, providing the “client” with the requested data.

Use the following object types with DDE:

- AI, AO, DI, DO
- DAT, MDAT
- DSC, TSC

SuperView 100 provides two types of data subscriptions using DDE, **one-time (demand) subscriptions** and **dynamic (cyclic) updated subscriptions**. A dynamic data update is done every 10 seconds by default, but this can be adjusted from the SuperView 100 Report software’s menu bar. Refer to Section 4.3.6 DDE Configuration, for further details.

See Section 3.1.3 Using Dynamic Data Exchange (DDE) for a tutorial explaining how to access process data from the ABB automation system.

Sending Data

You can send data to the ABB automation system using DSC and TSC. See Section 3.3.3, Using DDE to Access Process Data, for a tutorial explaining how to send data to the ABB automation system.

Carrying Out Commands in SuperView 100 Report Software

Use DDE to carry out commands to SuperView 100 Report software for report executions, etc. The DDE interface is identical to the command line interface, but has the added benefit that SuperView 100 Report software can be running at the time, which makes it much faster than the command line interface.

For more information about various commands, refer to Section 4.2.1.2 The Command Line Interface. See Section 3.1.3 Using Dynamic Data Exchange (DDE) for a tutorial explaining how to execute commands in SuperView 100 Report software using DDE.

DDE Syntax

See Appendix F Syntax for DDE Subscriptions, for a detailed description of DDE syntax.

3.1.4 The Remote File System (RFS) Server

You can transfer any record-oriented sequential file (ASCII or binary) from AS100 IMS to a remote node or from a remote node to AS100 IMS at the initiative of the remote node with the RFS Server. Files are transferred on a client-server basis.

The RFS Server has the following limitations:

- A maximum of 16 files can be open at one time.
- A maximum of four remote nodes can transfer files to this function at one time.
- The maximum file record size supported is 1,450 bytes.

Files transferred from SuperView 100 to a remote node must be placed in the RFS root directory (defined by the environment variable **SV_RFSDIR**) or any subdirectory below. Files received from a remote node will also be placed relative to this directory.

A description file is created for each file received. The description file contains information about the record structure of the transferred file and is needed to maintain the file structure when transferring to another node or when accessing the file with applications using the local file primitives. If no description file is found, default values are used.

NOTE

When the DOS version of the RFS server is active, the host computer is inaccessible for other duties.

3.1.5 The RFS File Transfer Library

The RFS file transfer library shall be used if you want to create your own RFS client application, such as the RFSFTU program (refer to Section C.3 RFS File Transfer Utility Program), in the DOS or MS-Windows environment.

The RFS file transfer library consists of two types of file-handling functions.

- Remote library functions
- Local file-handling functions

Remote library functions can be used to:

- Transfer any record-oriented sequential file (ASCII or binary) from AS100 IMS to a remote node and from a remote node to AS100 IMS.
- Translate logical report names to physical report names in MOD 300.
- Carry out TCL catalog functions on sequences and recipes in MOD 300.

These library functions free you from some of the cumbersome details involving message creation, transmission and receipt and communication with the remote system.

Local file-handling functions can be used to:

- Open, close, delete and create sequential files
- Read and write records from/to files
- Get file attributes or position the file pointer at the last record
- Create description files

These library functions streamline description file-handling for you in a variety of situations. For example, they can be useful when a file with a specific file structure is to be created in the local PC.

Only one user application can use the RFS file transfer library functions at one time, including the RFS Server. Refer to Appendix G RFS, for a more detailed description of the library functions.

3.2 Design Considerations

When you design a report, you must consider several things, including layout, to produce the best report possible. In the following sections, you will find hints on how to improve productivity when you are generating reports with SuperView 100 and spreadsheet programs.

3.2.1 Report Planning and Design

Use the following as a report planning and design checklist.

- What data is required for the report?
- How should this data be arranged for maximum clarity and ease of processing?
- What labels, lines, etc., should be inserted to improve the layout of the report?
- What calculations (formulas) are needed to obtain the desired results?
- What data should be presented as graphs, how should it be arranged and which types of graphs are suitable for use with this data?
- Will the report, or part of it, be used as a sub-report in another overview report?
- How should the report be output, to a disk file or to a printer?
- What degree of automation is needed for creation of the report? Should macros be used?

3.2.2 Report Generation Strategies

Several report generation strategies exist. Which one you choose depends mainly on the complexity of the report you want to produce. Spreadsheet software provides you with flexibility in retrieval, combination and production of reports in a variety of ways. The following sections contain brief discussions about various report generation strategies.

Formatted, Single Reports

There are two ways to produce simple reports containing labels, lines, formulas, etc.:

1. Include the formatting/calculation work in the report definition file.

   Use this method for simple reports with small amounts of data and few formatting requirements.

2. Use a special worksheet file which includes formatting/calculations.

   The benefit of this method is that you can use the same layout for a number of different reports. Choose it for more complex reports.

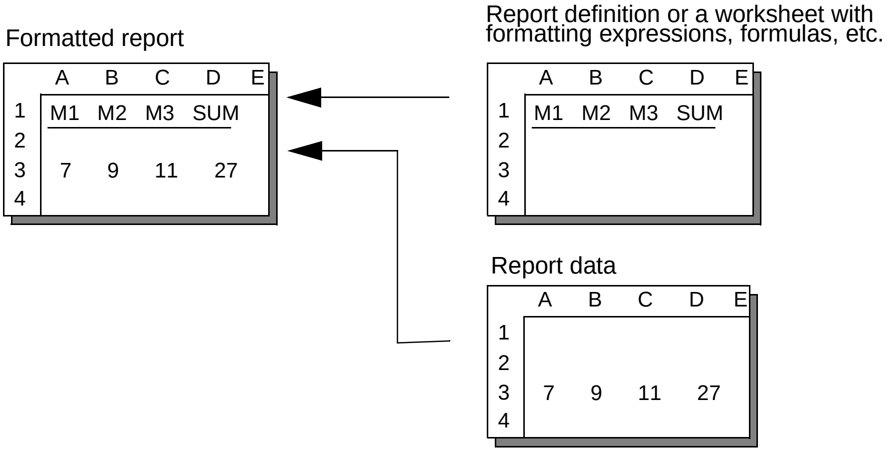

Formatted report
Report definition or a worksheet with formatting expressions, formulas, etc.
Report data
A B C D E
1 2 3 4
M1 M2 M3 SUM
7 9 11 27

Figure 3-8. A Formatted Single-Data Report Worksheet

Graphic, Single Reports

To create a graphic report, design the report definition with a graph referring to the desired data and create a suitable graphic layout.

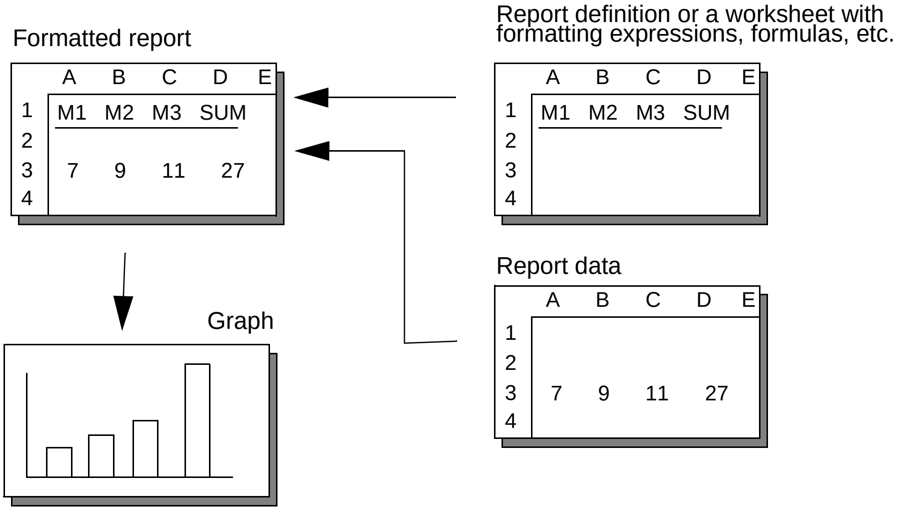


Figure 3-9. A Single-Graph Report Worksheet

Define and store frequently used or generated graphs in report definition worksheets or in report processing worksheets. Graph definitions should be carefully planned and tested, since the required data is not available in the report definition.

Layouts of the data ranges of the different types of report labels are described in detail in Appendix D Report Labels and Data Ranges.

Merged Reports

When merging several reports into a single summary or overview, define the formulas needed to process data from several reports in a separate report summary worksheet file.

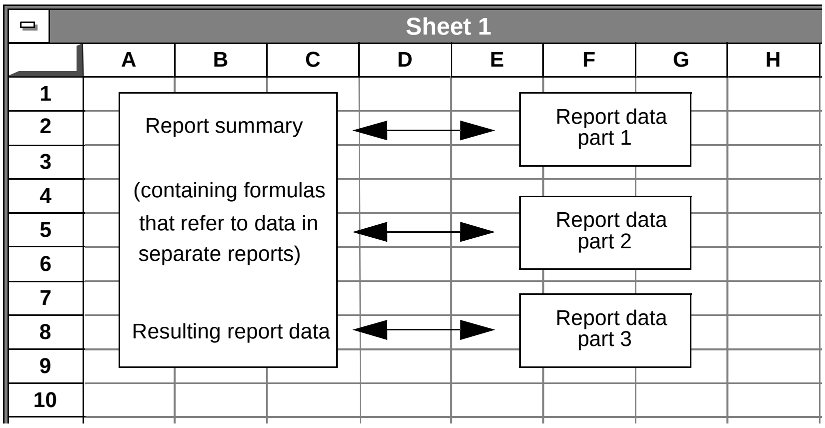
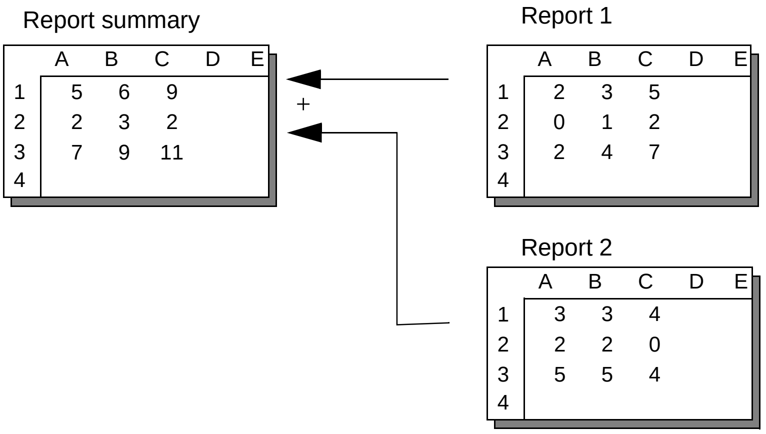


Figure 3-10. Multiple Report Summary Worksheet

The report summary worksheet is retrieved and the multiple report data worksheets are combined at different locations in the worksheet. The resulting data (calculated by formulas) will reside in a separate part of the worksheet, which makes up all the data of the summary report. Use the macro facility of the spreadsheet program to develop more complicated and/or repetitive commands.

Data-Summary Reports

One way to create reports is to sum up cell data from different reports. The result is a summary of the data in all cells across all the reports.



Report summary

	A	B	C	D	E
1	5	6	9		
2	2	3	2		
3	7	9	11		
4					

Report 1

	A	B	C	D	E
1	2	3	5		
2	0	1	2		
3	2	4	7		
4					

Report 2

	A	B	C	D	E
1	3	3	4		
2	2	2	0		
3	5	5	4		
4					

Figure 3-11. A Data Summary Report Worksheet

A simple way you can compare data in two different reports is to subtract the cell data in one report from those of another.

Multiple-Layout Reports

Create the report definition file using only report labels when you create advanced applications. This gives you the flexibility to use other worksheet or macro files containing the report formatting and calculations to produce different report layouts based on the same data or to produce reports with different data using the same report layout.

When you are ready to generate a report, retrieve the report definition worksheet you have chosen and combine it with the report data you wish to use to create the desired report.

Using Formulas in Reports

Use the spreadsheet program's built-in formulas to calculate report data based on retrieved report data. Enter these formulas into the report definition file or the formatting file in the same way you would any other worksheet file.

Enter these formulas carefully to ensure that report data is correctly referenced since the required data is not available in the report definition. Layouts of the data ranges of the different types of report labels are described in detail in Appendix D Report Labels and Data Ranges.

Formulas are particularly helpful when you use bitsets, for example, when you use DSC objects containing digital values in reports. In such a case, digital values are represented by bits in data cells of bitsets. These values can be extracted by formulas.

Error-Handling in Reports

The formulas that process report data can include error-handling for the report application in two basic ways.

1. The error labels that appear when results are out of range can be detected and appropriate action can be taken.
2. Default data can be entered at strategic locations so that lack of data can be detected or prevented from disrupting calculations.

For a description of the error labels that may appear as a result of data collections, see Section 5.3.1 Error Labels.

Using Macros in Reports

Simplify report generation by using the macro facility offered by the spreadsheet program. Macros automate repetitive or complicated command sequences, which increases productivity and decreases the risk of errors.

Include macros to perform the following tasks:

- Automate long and repetitive command sequences.
- Incorporate report data into report worksheets.
- Format single data or ranges of data.
- Extract, restructure and/or group data.
- Present quick graphs of data.
- Merge multiple report data worksheets.
- Automate the outputting of reports.

A single-step macro-execution feature is normally incorporated in the spreadsheet for debugging when you create macros. To use macros for more advanced purposes, refer to your spreadsheet software’s documentation on the subject.

3.3 Capacity and Performance

3.3.1 SuperView 100Report Software

The following table represents the total time you will need to process a report request for data from resolved objects in ABB Automation system nodes and to create a report data file:

Table 3-1. SuperView 100 Report Program Performance

Description	Time (sec.)
20 DSC (480 data values)	2
20 TSC (20 text strings)	2
20 AI or AO or DI or DO or DAT	2
20 MDAT (1000 data values)	2
20 TTD (4800 data values)	5
Mixed data (3190 data values)	4
Mixed data (6380 data values)	8

These times were obtained on a Compaq® Deskpro® 386/20, with a 130 MB disk and default configuration under normal conditions.

This test was made with the MS-Windows version of SuperView 100, but expect similar results for the DOS version. These execution times, however, depend on the type of machine as well as on the disk access time and the network load.

Acquisition Capacity

The number of data values that can be retrieved for a report object depends on the type of object/label.

Table 3-2. Number of Data Values Fetched for Each Object Type

Object type	Data (max.)
DSC	24 values
TSC	1 text string
MDAT	50 values
DAT	1 value
AI and AO	1 value
DI and DO	1 value
TTD and TTDO	1000 values
EVL	600 events
EVLC	599400 events
DATE and TIME	1 value
TRIGG	24 values or 1text string

3.3.2 The Remote File System

The following table represents the total time needed for the RFS to transfer a record-oriented sequential ASCII file between two SuperView 100 systems on a GCOM network with variable package size.

Table 3-3. RFS Performance

File size	Time (sec.)
1 kB	1
8 kB	3
16 kB	5
128 kB	26
512 kB	100

These times were obtained between two Compaq Deskpro 386/20, with a 130 MB disk and default configuration under normal conditions.

This test was made with the MS-Windows version of SuperView 100, but similar results can be expected for the DOS version. These execution times, however, depend on the type of machine as well as on the disk access time and the network load.

3.3.3 Primary Memory

SuperView 100 requires primary memory for software and data storage during report requests and file transfers as follows.

For the **MS-Windows** version:

- The SuperView 100 device driver for GCOM communication needs approximately 64 kB of the DOS RWM memory when installed.
- To execute programs under MS-Windows, at least 2 MB primary memory is needed. SuperView 100 must also allocate a buffer of 1.5 kB of DOS real mode memory to communicate with the GCOM driver from Windows.

For the **MS-DOS** version:

- The SuperView 100 device driver for GCOM communication needs approximately 64 kB of memory when installed.
- The SuperView 100 RFS Server needs approximately 60 kB of memory for code during file transfers.
- The SuperView 100 RFS File Transfer Utility program needs approximately 65 kB of memory for code during file transfers.
- These programs allocate a dynamic data area of about 8 kB during file transfers.
- The SuperView 100 RFS Configuration program needs approximately 15 kB of memory for code during execution.
- The SuperView 100 Report software needs approximately 110 kB of memory for code during report executions.
- The SuperView 100 Report software allocates a data area of approximately 15 kB during execution. In addition, the program allocates a dynamic data area, the size of which depends on the type and number of labels in the report.

Table 3-4. Primary Memory Allocated for Different Object Types

Object type	Allocates max. (bytes)
DSC	164
TSC	162
MDAT	268
DAT	74
AI and AO	94
DI and DO	74
TTD and TTDO	4,098
COM	664
EVL and EVLC	50,400
DATE and TIME	0
TRIGG	164

3.3.4 Disk Storage

You will need disk storage for all report definitions, report data, report result files and files transferred with RFS. The complete SuperView 100 package allocates approximately 1 MB of disk storage when installed.

Each report requires additional disk memory for:

- The report definition file
- The symbol file
- The TTD symbol file
- The report data file
- The report result file

The amount of disk memory needed for each report definition file depends on formulas, labels and graphs for formatting of the report. Approximately 8.5 kB of disk memory are needed to store a report definition file containing 160 report object definitions of mixed types without any formula macros or other worksheet definitions.

The amount of disk memory needed for symbol files depends on the total number of unique AI, AO, DI, DO, DAT, MDAT and TTD objects in the report definition files. Each unique object needs 30 bytes in the symbol file. If TTDO labels are used for TTD objects, each unique TTD object needs an extra 40 bytes in the TTD symbol file.

Approximately 3.6 kB of disk memory are needed to store the symbols needed for a single report definition file containing 160 report object definitions of evenly mixed types.

The report data files need 10 bytes for header information. In addition, the report data files need the storage capacity listed in the following table, depending on the object types in the report. The data for each object/label requires the following amount of storage space:

Table 3-5. Disk Storage Needed for Each Object Type

Object type	Disk needed (bytes)
DSC	18 - 408
TSC	64 - 134
MDAT	18 - 850
DAT	52
AI and AO	88
DI and DO	34
TTD and TTDO	170 - 17,152
COM	52 - 550
EVL and EVLC	232 - 138,600

Table 3-5. Disk Storage Needed for Each Object Type

Object type	Disk needed (bytes)
DATE and TIME	18
TRIGG	18 - 408

Approximately 220 kB of disk memory are needed to store a report data file containing data for 160 objects of mixed types (approximately 12,500 data values). Approximately 140 kB of disk memory are needed to store a report data file from the EVL or EVLC function containing 600 process events.

3.4 Software Setup

After installation of the SuperView 100 software, the PROTOCOL.INI file must be manually updated with binding between the GCOM device and the IEEE 802.3 board driver. If the SuperView 100 installation created a new PROTOCOL.INI file, it must manually be updated with the IEEE 802.3 board driver. You must also check that the IEEE 802.3 board driver is loaded, after the NDIS protocol manager driver (PROTMAN), in the CONFIG.SYS file. Please refer to your IEEE 802.3 board documentation.

If you did not let the installation program update your AUTOEXEC.BAT and CONFIG.SYS files with the correct setup information, also these files must be modified before the SuperView 100 software can be started. Please refer to Section 2.2 Setup for further information about how these files must be setup. Also note that the PC must be re-booted to activate any changes made in these files.

3.5 Tutorials

You will find the following tutorials in this chapter, each providing step-by-step instructions for a SuperView 100 function:

- **Report Production with Spreadsheet Software**

  This tutorial shows you how to create a complete report using spreadsheet software.

- **Using Batch Files for Report Sheduling**

  You can use SuperView 100 batch files for simple report sheduling. This tutorial describes how you can perform different schedule functions.

- **Using DDE to Access Process Data**

  The DDE tutorial describes how you can request data from Microsoft Excel and send it to the ABB automation system using the SuperView 100 Report software DDE interface

All examples use Microsoft Windows and Excel, but most file format compatible spreadsheet software can perform the same functions.

3.5.1 Report Production with Spreadsheet Software

In this section, you will take a guided tour through report production with SuperView 100 and Microsoft Excel. Step-by-step instructions for creating an example of a simple production report comprised of one **MultiDAT (MDAT)** and one **Data Set (DSC)** follow.

Example Background

The ABB automation system configuration for the example used in this tutorial is shown in the following figure.

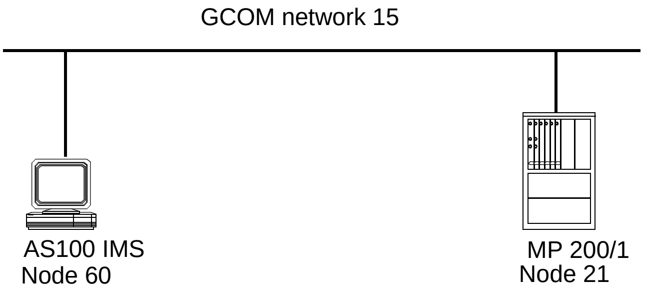


Figure 3-12. System Configuration Used in Tutorial

The AS100 IMS, with SuperView 100 software, is connected as Node 60 on the GCOM network 15. The communication setup parameters in the SuperView 100 AUTOEXEC.BAT file for this configuration become:

```
SET NETW=15
SET NODE=60
```

The sample report collects DSC and MDAT data from the MP 200/1 as follows:

- An MDAT consisting of four real-value DATs holding the production figures for four different production lines for the most recent 24 hours.
- A DSC sending a DS with identity 11, containing four Boolean DATs and four real-value DATs representing the status and the current flow through these production lines.

The MP 200/1 Application

The MP 200/1 must have one DS with identity 11, sending data to IMS 100. The DS should contain four Booleans and four real-value DATs. The DS must be defined as follows,

Table 3-6. Data Set Definition in the MP 200/1

Property	Value
Name	FLOW.14
DS identity	11
Number of Bools	4
Number of Int	0
Number of Intl	0
Number of Real	4
Type	Sending
Netw	15
Node	60

where:

- The four Booleans hold the status of each production line, with the value of bit 0 (LSB) representing the On/Off status.
- The four real-value data items hold the current flow rates through each line.

NOTE

The DS must be **ACTIVE**, but also **BLOCKED**. (No data is sent prior to a data request.)

The MP 200/1 must also have a MDAT containing four consecutive, real-value DAT records, the first named **PROD24.1**. The MDAT must refer to consecutive DAT objects according to the following,

Table 3-7. MultiDAT Definition in the MP 200/1

Name	PROD24.1	PROD24.2	PROD24.3	PROD24.4
Value	23.7	56.2	34.6	41.3

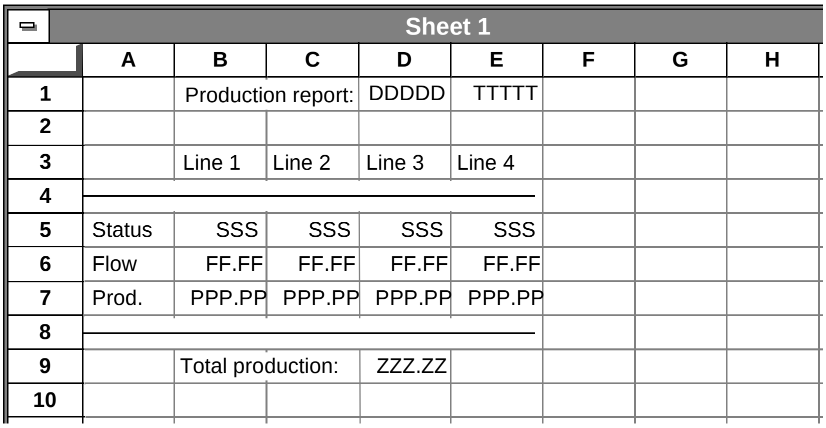
where the four DATs represent production during the most recent 24 hours of the four production lines.

NOTE

DATs of MDATs must be consecutive records in the MP 200/1 database.

Planning the Report

The first step in producing a report is planning the contents and layout of the report worksheet and the layout and formatting of the report output. The report layout for the sample report will look like the example below:



	A	B	C	D	E	F	G	H
1		Production report:		DDDDD	TTTTT			
2								
3		Line 1	Line 2	Line 3	Line 4			
4								
5	Status	SSS	SSS	SSS	SSS			
6	Flow	FF.FF	FF.FF	FF.FF	FF.FF			
7	Prod.	PPP.PP	PPP.PP	PPP.PP	PPP.PP			
8								
9		Total production:		ZZZ.ZZ				
10								

Figure 3-13. The Desired Report Layout

Report data is distributed as follows in the report layout:

- **DDDDD** in cell D1 represents the date of the report.
- **TTTTT** in cell E1 represents the time of the report.
- **SSS** in cells B5-E5 should be **On** or **Off** depending on the value of bit 0 (LSB) of status.
- **FF.FF** in cells B6-E6 represents the current flow.
- **PPP.PP** in cells B7-E7 represents the latest 24-hour production.
- **ZZZ.ZZ** in cell D9 represents the calculated sum of the latest 24-hour-production figures from the four production lines.

Report formatting appears in the report layout as follows:

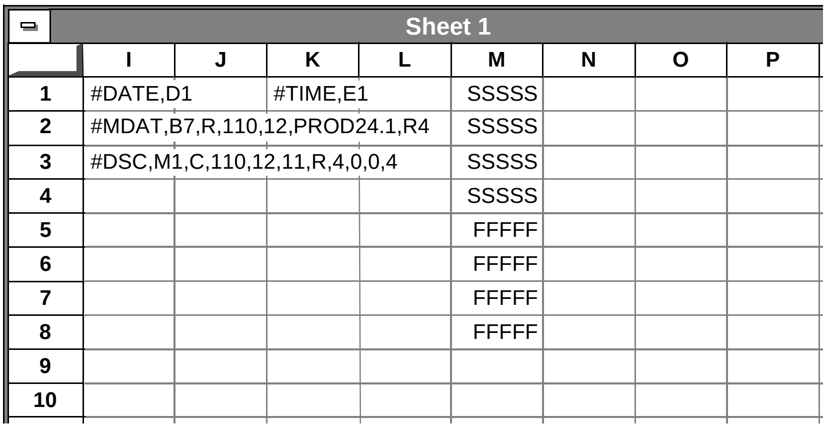
- The **Production report:** label in cell B1.
- The **Line x** right-adjusted labels in cells B3-E3.
- The **---- ... ----** dividing line labels in cells A4 and A8.
- The **Status**, **Flow** and **Prod.** labels in cells A5-A7.
- The **Total production:** label in cell B9.

Report labels include:

- One **DATE** report label for the date of the report.
- One **TIME** report label for the time of the report.
- One **DSC** report label for status and the current flows.
- One **MDAT** report label for 24-hour production figures.

Defining the Report

The next step in defining the report is entering the necessary report labels. Normally, report labels should be defined somewhere other than the report since they will not be part of the final report. The report labels are located in the **definition area**.



	I	J	K	L	M	N	O	P
1	#DATE,D1		#TIME,E1		SSSSS			
2	#MDAT,B7,R,110,12,PROD24.1,R4				SSSSS			
3	#DSC,M1,C,110,12,11,R,4,0,0,4				SSSSS			
4					SSSSS			
5					FFFFF			
6					FFFFF			
7					FFFFF			
8					FFFFF			
9								
10								

Figure 3-14. The Report Labels Defined in the Worksheet

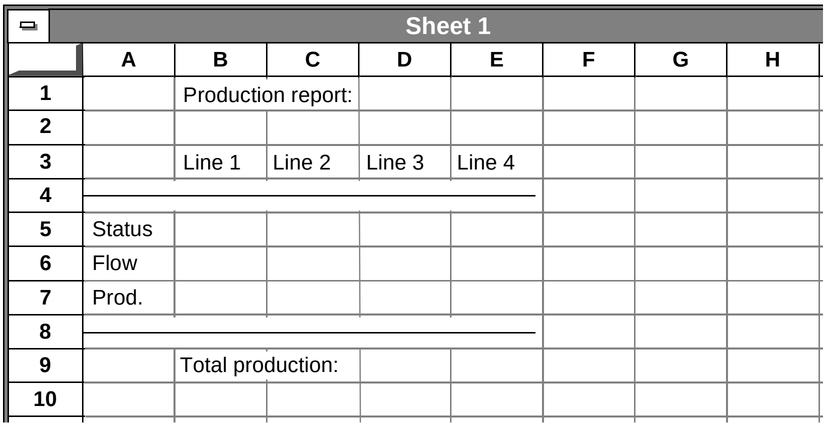
The MDAT label puts the production data directly into cell range B7-E7 in the report layout, which is aligned by rows. The MDAT label collects four consecutive real-value DATs from the MP 200/1, starting with PROD24.1.

The layout of the data ranges for status and flow in the DSC must be converted and put into the report. For this reason, it is stored in the definition area. The data range will be inserted by column, to the right of the report labels, starting in cell M1. The DS has the identity 11, is of the **receive** type, and contains four Booleans and four real values.

Report labels define where the data ranges (areas for the collected data) will reside in the worksheet. The following data ranges will be used.

- Data ranges in the definition area (starting at cell I1):
  - DSC line status in cells M1-M4 (marked **SSSSS**).
  - DSC line flow in cells M5-M8 (marked **FFFFF**).
- Data ranges in the report area (starting at cell A1):
  - DATE (the date of the report request in cell D1).
  - TIME (the time of the report request in cell E1).
  - MDAT (production data in cells B7-E7).

The next step in producing a report is entering the necessary report formatting and formulas. Move back one screen to the left on the worksheet to the report layout area and enter formatting labels as follows:



Sheet 1

	A	B	C	D	E	F	G	H
1		Production report:						
2								
3		Line 1	Line 2	Line 3	Line 4			
4								
5	Status							
6	Flow							
7	Prod.							
8								
9		Total production:						
10								

Figure 3-15. The Worksheet with Formatting Labels

Entry of Report Formula

Enter the formula for collection and conversion of data from the definition area and a formula for calculating the total production next.

1. Enter the formula for total production first. Using the Excel function **SUM()**, the formula will sum up all production results in cells B7-E7 and put the sum into cell D9. It looks like this in Microsoft Excel:

   =SUM(B7:E7)

2. Next, transfer the data for the four flow values from cells M5-M8 to cells B6-E6 by making simple references to the cells in the definition area from the cells in the report. The contents of cell M5 must be copied to cell B6, the contents of cell M6 must be copied to cell C6, etc. For example:

   B6=M5

3. Transfer and convert the status of the four lines from cells M1-M4 to cells B5-E5. In this transfer, convert the labels to On or Off status, depending on the value of the least significant bit (bit 0) of the status value. The value of the least significant bit of a cell is easily obtained with the Excel MOD function using the formula **MOD(cell,2)**.

   It is easy to create text strings on the basis of tests of data values using the **IF** function. The complete formula to use in case of On/Off status is **"=IF(MOD(cell,2), "On", "Off")"**. The contents of cells M1 to B5, M2 to C5, etc., must be converted.

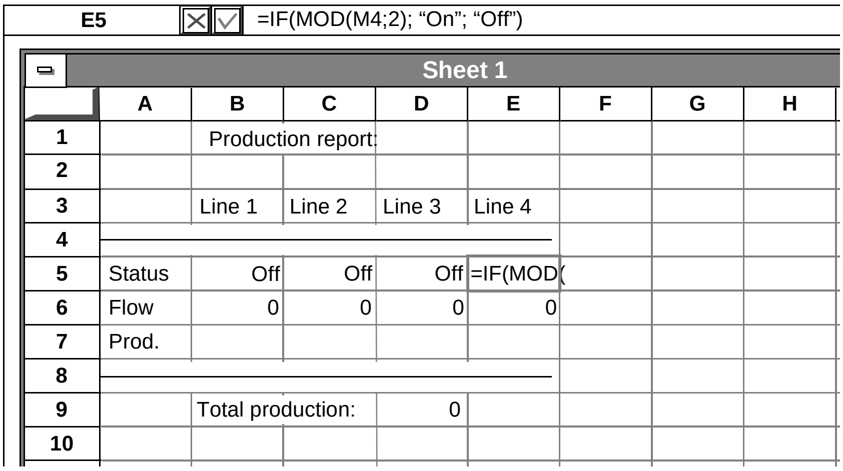
   Enter the following formulas in cells B5 to E5,

   =IF(MOD(*x*;2),"On","Off")

   where:

   x=M1 to M4

Once you have entered these report formulas, the worksheet screen should look like this:



E5 =IF(MOD(M4;2); “On”; “Off”)

Sheet 1

	A	B	C	D	E	F	G	H
1		Production report:						
2								
3		Line 1	Line 2	Line 3	Line 4			
4								
5	Status	Off	Off	Off	=IF(MOD(			
6	Flow	0	0	0	0			
7	Prod.							
8								
9		Total production:		0				
10								

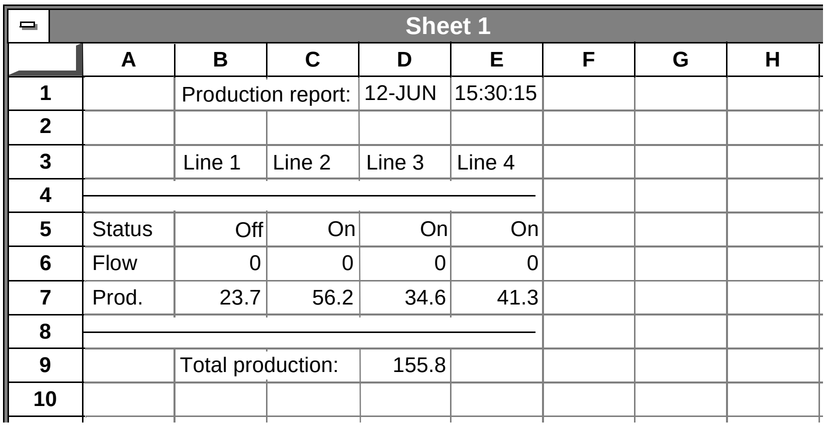
Figure 3-16. The Worksheet Complete with Formulas

The report definition worksheet is now complete. Save it to a worksheet file so it can be used as the basis for report requests.

Requesting the Report

You can request the report now the report definition worksheet file has been created. Call up SuperView 100 Report software using the Windows interface, DOS command line or a macro definition in the spreadsheet software (for example using the Windows interface choose **Report exec** from the **File** menu. Select the report definition file from this dialog box and press **OK)**.

Retrieve the report definition file after the report execution and combine it with the report data file. The report is complete and the worksheet screen should look something like this:



	A	B	C	D	E	F	G	H
1		Production report:		12-JUN	15:30:15			
2								
3		Line 1	Line 2	Line 3	Line 4			
4								
5	Status	Off	On	On	On			
6	Flow	0	0	0	0			
7	Prod.	23.7	56.2	34.6	41.3			
8								
9		Total production:		155.8				
10								

Figure 3-17. The Final Report

Outputting the Report

You now have access to the full power and flexibility of spreadsheet software to output the report to a printer or to disk.

3.5.2 Using Batch Files For Report Scheduling

You can use the SuperView 100 batch files interface for simple report sheduling. In this tutorial, you will see an example of how to generate a cyclic report production.

Planning the Batch File

The report used for this sample batch file has three terminate triggers to achieve a number of different report executions, depending on which trigger has terminated.

Defining the Batch File

You can define a the batch file using any text editor, such as MS-Windows NotePad. In this example, the batch file might look as follows:

```
:START
SV100 TRIGGREPORT.XLS
IF TRIGGIDENT=1
GOSUB TRIGG1
IF TRIGGIDENT=2
GOSUB TRIGG2
IF TRIGGIDENT=3
GOSUB TRIGG3
GOTO START

:TRIGG1
SV100 REPORT1.XLS
SV100 REPORT2.XLS
RETURN

:TRIGG2
SV100 REPORT3.XLS
SV100 REPORT4.XLS
RETURN

:TRIGG3
SV100 REPORT5.XLS
SV100 REPORT6.XLS
RETURN
```

Figure 3-18. The Batch File used in Tutorial

The batch file execution restarts from the beginning when the trigger treatment is finished. Name the batch file with extension **.SV2** so it will be automatically listed in the Batch File Selection dialog box. Refer to Section 4.3.1 Batch File Execution, for further information.

For a complete description of the various functions you can be use in SuperView 100 batch files, see Section 3.1.2 Report Scheduling with SuperView 100 Batch Files.

Batch File Execution

Start the batch file using the SuperView 100 Report software Windows interface. For a more complete description, see Section 4.3.1 Batch File Execution.

3.5.3 Using DDE to Access Process Data

The SuperView 100 DDE interface is an easy way to access process data from other MS-Windows applications supporting DDE. In this example, we are using Microsoft Excel to do the following:

- Request data for an AI object.
- Send a DS.
- Carry out a command in SuperView 100 Report software.

Dynamic Request for Process Data

Use **warm DDE links** to request process data with dynamic updating. This process data will then be cyclically updated. The cycle time can be adjusted between 5 and 60 seconds from SuperView 100.

To create warm DDE links using Microsoft Excel:

1. Mark the cells where the data is to be placed.
2. Enter the DDE link definition.

NOTE

Data will only be displayed in the cells marked. If four values are expected and only two cells are marked, the last two values will not be displayed.

When all of the values are needed for the AI object, mark 5 cells and enter the following DDE link definition:

=SV100|AI!AIC21_1\ FULL

where SV100 refers to the application name, AI is the topic name referring to the object type and the item string contains the object name (AIC21_1) followed by \FULL which refers to the data item to be collected, in this case all items. For more information about the DDE syntax, please refer to Appendix F Syntax for DDE Subscriptions.

The spreadsheet will now look like this:

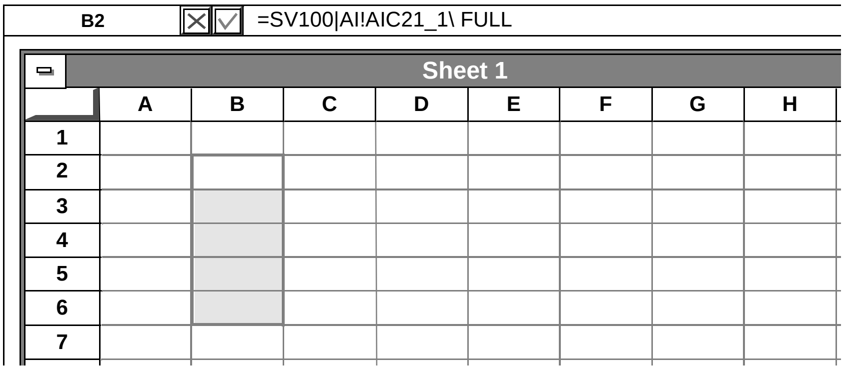
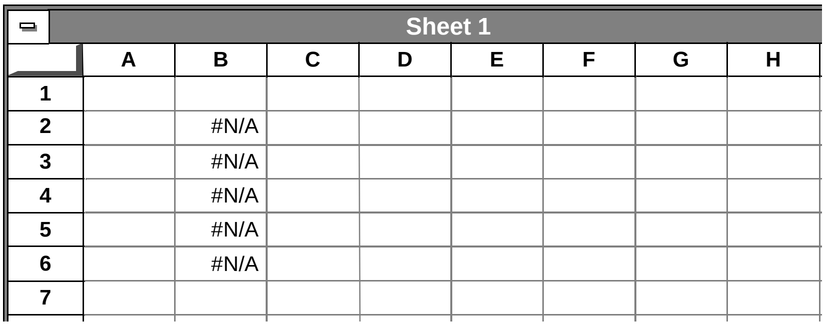


Figure 3-19. DDE Link Definition used in Tutorial

NOTE

The DDE link definition must be followed by **<CTRL>+<SHIFT>+<ENTER>** in Microsoft Excel to achieve array structure of the data.

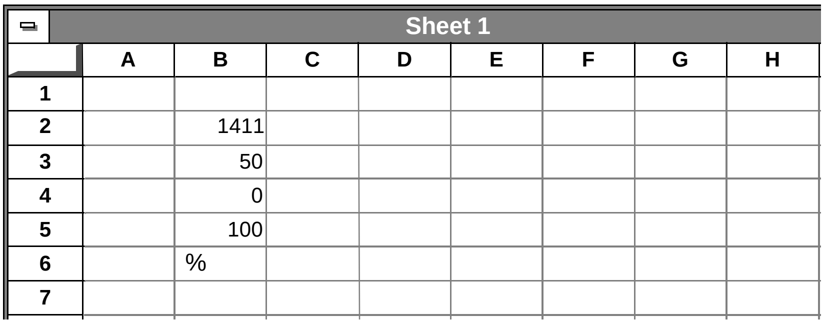
When the command is activated, it will take a while before the data is received. Meanwhile, the cells will be marked with **#N/A** as follows:



	A	B	C	D	E	F	G	H
1								
2		#N/A						
3		#N/A						
4		#N/A						
5		#N/A						
6		#N/A						
7								

Figure 3-20. DDE Link Waiting for Data

Data will, after a short time, be displayed in the selected area as follows:



	A	B	C	D	E	F	G	H
1								
2		1411						
3		50						
4		0						
5		100						
6		%						
7								

Figure 3-21. Result of DDE Link

The numbers displayed in the figure above represent the following:

- **1411** is the *status* of the AI object.
- **50** is the object *value.*
- **0** and **100** are the *rmin* and *rmax* levels, respectively.
- **%** is the engineering *unit*.

For more information about the DDE syntax for different object types, see Appendix F Syntax for DDE Subscriptions.

Demand Request for Process Data

Use **cold DDE links** for demand requests for process data. The DDE link is terminated after the process data is fetched. The DDE link must be defined again to update this data.

Cold DDE links are created from Microsoft Excel using the following DDE functions in a macro definition sheet:

1. *channel_num* **=INITIATE**(*app_text, topic_text*)

   Open a DDE channel to the application.

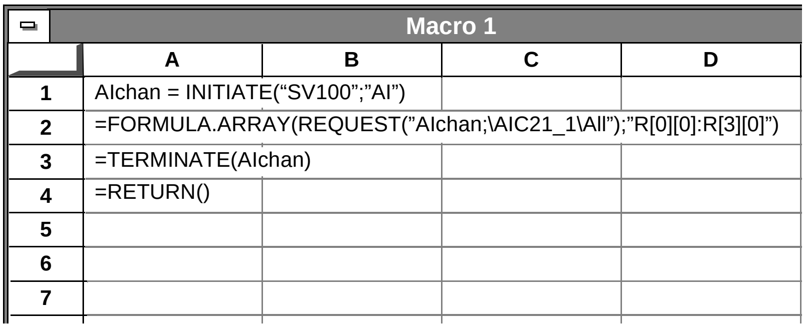
2. **=REQUEST**(*channel_num, item_text*)

   Request for the data.

3. **=TERMINATE**(*channel_nuum*)

   Close the DDE channel.

A macro to fetch the same values as given in the previous example, with data displayed by column starting at the current cursor position in the worksheet, will look like this:



	A	B	C	D
1	AIchan = INITIATE(“SV100”;”AI”)			
2	=FORMULA.ARRAY(REQUEST(”AIchan;\AIC21_1\All”);”R[0][0]:R[3][0]”)			
3	=TERMINATE(AIchan)			
4	=RETURN()			
5				
6				
7				

Figure 3-22. Macro Sheet for a DDE Demand Request

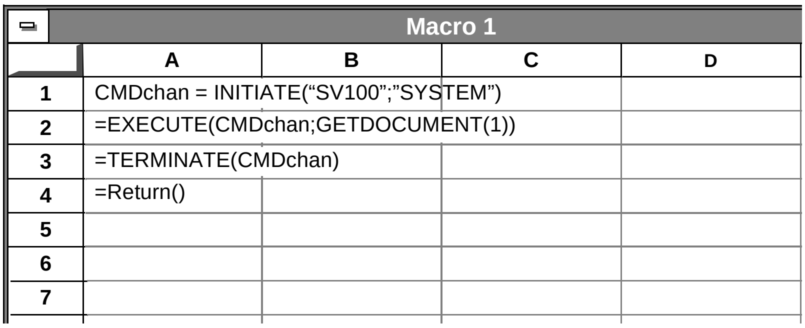
Carry Out Commands in SuperView 100 Report Software

You can also carry out commands in SuperView 100 Report software using DDE. This interface is identical to the command line interface, except SuperView 100 Report software offers the added benefit that is running when the macro execution starts. Performance is improved because the DDE call does not have to start up the SuperView 100 software.

A command is carried out in another application from Microsoft Excel using the following DDE function:

=EXECUTE(*channel_num, execute_text*)

A macro to execute the current report definition, opened in Microsoft Excel, might look like this:



	A	B	C	D
1	CMDchan = INITIATE(“SV100”;”SYSTEM”)			
2	=EXECUTE(CMDchan;GETDOCUMENT(1))			
3	=TERMINATE(CMDchan)			
4	=Return()			
5				
6				
7				

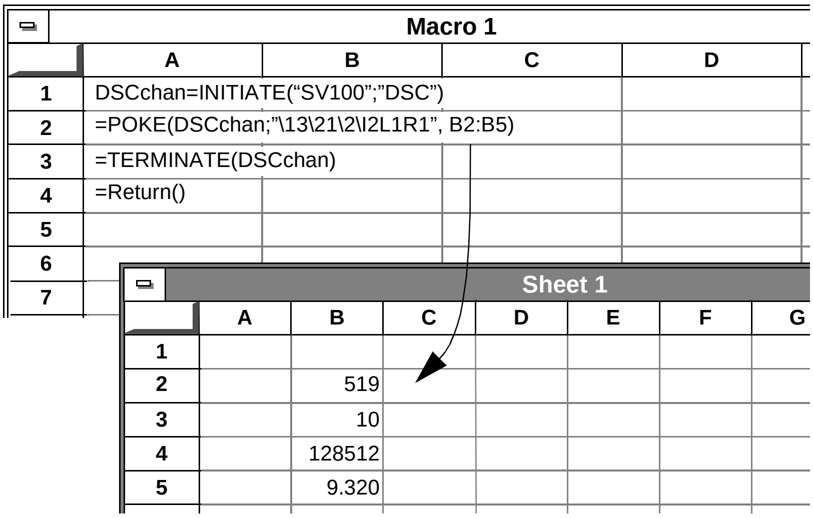
Figure 3-23. Macro Sheet for DDE Command Execution

Send Data to the ABB Automation System

You can send data back to the ABB automation system using SuperView 100 and the DDE **POKE** function. The POKE function has the following syntax:

=**POKE**(*channel_num, item_text, data_ref*)

To create a sample macro, send DS data to a MP 200/1 with **netw = 13** and **node = 21**. The DS identity is 2 and consists of four values (two integer, one integer long and one real value). The macro might look as follows:



Macro 1

	A	B	C	D
1	DSCchan=INITIATE(“SV100”;”DSC”)			
2	=POKE(DSCchan;”\13\21\2\I2L1R1”, B2:B5)			
3	=TERMINATE(DSCchan)			
4	=Return()			
5				
6				
7				

Sheet 1

	A	B	C	D	E	F	G
1							
2		519					
3		10					
4		128512					
5		9.320					

Figure 3-24. Macro Sheet for Transmission of Data with DDE

When the macro is executed from **sheet1** (sheet1 as the active window), the data in cells B2 to B5 is sent.

NOTE

Data types specified in the DSC item string must always be specified in the correct order: Boolean (B), integer (I), long integer (L) and real (R). However one or more of the data types can be omitted as in the example above.

3.6 Menus

The user interface for report definitions is the spreadsheet software. For more information on menus, please refer to your spreadsheet software’s documentation.

For a description of the SuperView 100 user’s interface refer to Section 4.2 Product Operation.

Chapter 4 Runtime Operation

4.1 Operating Overview

This chapter describes the operation of SuperView 100 in the MS-Windows environment. See Appendix C MS-DOS Version Specific, for an in-depth description of the MS-DOS version of SuperView 100.

SuperView 100 Report software and the RFS Server are represented by the following icons in the SuperView 100 program group:

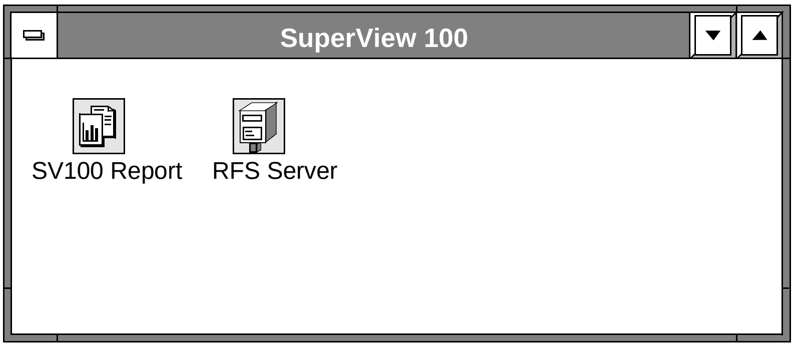


Figure 4-1. SuperView 100 Program Group in MS-Windows

This group is automatically generated by the SuperView 100 installation program. Double-click on the program icons to start these programs, as you would other MS-Windows applications.

4.2 Product Operation

The MS-Windows version of the SuperView 100 Report software and the RFS server can execute simultaneously with other MS-Windows applications, but cannot coexist with one another. Terminate SuperView 100 Report software before you execute the RFS server, and vice versa.

4.2.1 SuperView 100 Report Software

There are three user interfaces for the SuperView100 Report program in MS-Windows:

- Windows user
- Command line
- DDE

The Windows user interface provides convenient services to Windows users for interactive report executions and for report scheduling using a special type of batch file supported by SuperView 100.

The command line interface can be used for interactive report compilation on command from other MS-Windows applications, such as macros defined in spreadsheet software.

The DDE interface can easily be used by other MS-Windows programs supporting DDE to read or send process and calculated data from/to the rest of the automation system. This interface also provides command execution in the SuperView 100 Report software, similar to the command line interface.

4.2.1.1 The Windows Interface

The main window, below, is displayed on the screen when you start the SuperView 100 Report program from MS-Windows:

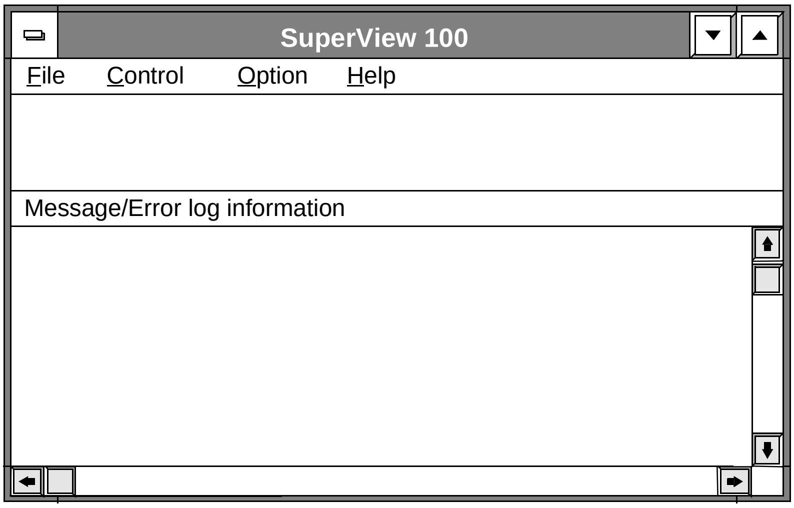


Figure 4-2. SuperView 100 Report Program Main Window

The File Menu

Use the **File** menu to select a batch file or report to execute or to exit SuperView 100. Choose **File** from the main window menu bar. The following drop-down menu is displayed:

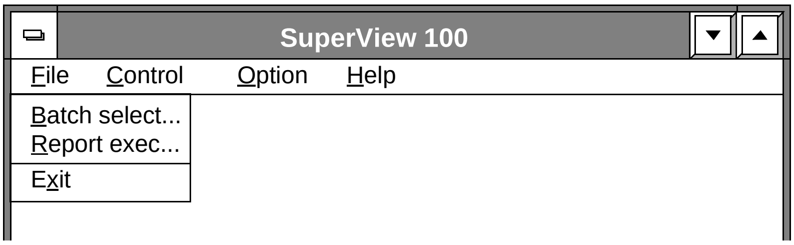


Figure 4-3. The File Menu

- Choose **Batch select** to select a batch file to execute.
- Choose **Report exec** to select a report to execute.
- Choose **Exit** to exit the SuperView 100 Report program.

The Control Menu

Use the **Control** menu for batch file and report execution control. Choose **Control** from the main window menu bar. The following drop-down menu is displayed:

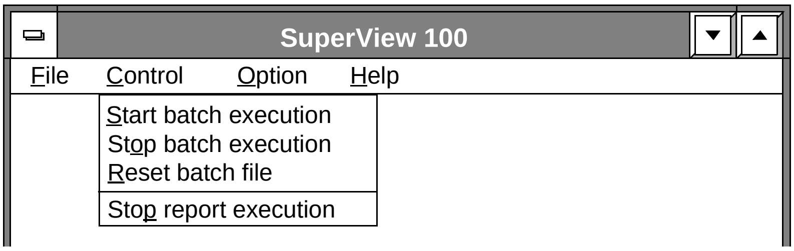


Figure 4-4. The Control Menu

- Choose **Start batch execution** to start executing a selected batch file. If you stop the batch execution, you can continue the batch execution where it left off.
- Choose **Stop batch execution** to stop a batch file execution.
- Choose **Reset batch file** to reset the batch file to the first line after the batch file execution has been stopped.
- Choose **Stop report execution** to terminate the current report execution before it is completed. You may need this option when executing reports using TRIGG or EVLC labels.

The Option Menu

Use the **Option** menu to choose a number of SuperView 100 Report software utilities. Choose **Option** from the main window menu bar. The following drop-down menu is displayed:

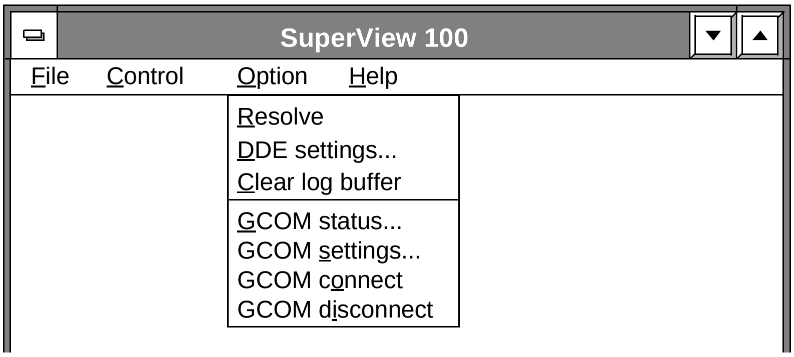


Figure 4-5. The Option Menu

- Choose **Resolve** to resolve, or translate, the symbols and TTD symbols in the SuperView 100 symbol files.
- Choose **DDE settings** to configure cyclic DDE requests.
- Choose **Clear log buffer** to clear the messages in the log window.
- Choose **GCOM settings** to configure GCOM.
- Choose **GCOM status** to display the GCOM device status.
- Choose **GCOM connect** to reconnect SuperView 100 to the GCOM network after a GCOM disconnect.
- Choose **GCOM disconnect** to disconnect SuperView 100 from the GCOM network.

Help Menu

Use the **Help** menu when you need help with, or additional information about, SuperView 100 Report software. Choose **Help** from the main window menu bar. The following drop-down menu is displayed:

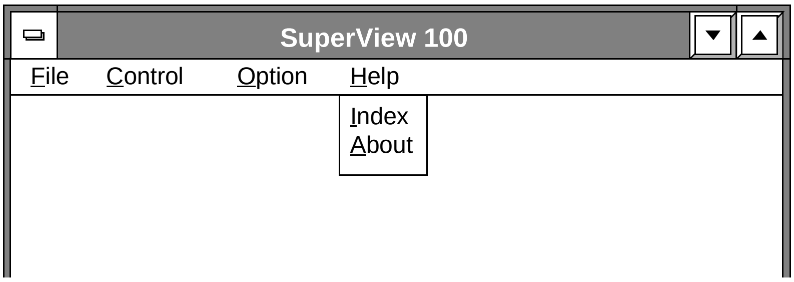


Figure 4-6. The Help Menu

- Choose **Index** to access help information about any of the SuperView 100 Report software functions.
- Choose **About** to display the current version of the SuperView 100 software.

4.2.1.2 The Command Line Interface

The SuperView 100 Report program takes command line parameters as qualifiers to define program execution. To use the command line interface, make sure that the SuperView 100 Report program is not already executing because the command line interface will attempt to start a new instance of the program. This is not allowed. When the command, for example, a report execution, is completed, the SuperView 100 Report program automatically terminates.

To send commands to the SuperView 100 Report program when it is already running, use the DDE interface. See Section 3.1.3 Using Dynamic Data Exchange (DDE), for further information.

Syntax

The syntax of the command line for the SuperView 100 Report program is as follows:

SV100WIN [*drive:][path]filename.ext [qualifiers*]

The following command line qualifiers are recognized by the SuperView 100 Report program:

- **/STATUS**

  Display the GCOM subnet status on the screen.

- **/DISCONNECT**

  Disconnect from the GCOM subnet after execution.

- **/RESOLVE**

  Resolve all symbols defined in the symbol file and all the TTD symbols in the TTD symbol file.

- **/OUTFILE=<*file*>**

  Put data output into the file named <*file*>.

You can abbreviate the qualifiers by leading character(s), for example, **/D** or **/DIS** for disconnect.

Status Qualifier

The **/STATUS** qualifier displays the status of the GCOM subnet on the screen. See Section 4.3.5 GCOM Status, GCOM Status, for more information.

The transit node table, appearing in the status display, provides important information. It will list the connected and active nodes in the GCOM subnet. Use the status qualifier alone or with other parameters and qualifiers. When you use it in combination with other parameters and qualifiers, the status display is shown before any other actions are taken.

Disconnect Qualifier

The **/DISCONNECT** qualifier disconnects SuperView 100 from the GCOM subnet. Use the qualifier alone or with other parameters and qualifiers. When you use it in combination with other parameters and qualifiers, /DISCONNECT is executed last.

Resolve Qualifier

The **/RESOLVE** qualifier translates all symbols in the **GCOMSYMB.DTA** and **TTDSYMB.DTA** symbol files. If symbols or TTD symbols are found that cannot be translated, they are written to symbol error files named **GCOMSYMB.ERR** or **TTDSYMB.ERR**, respectively.

Use the /RESOLVE qualifier alone or with other parameters and qualifiers. When you use it in combination with other parameters and qualifiers, /RESOLVE is executed before all other qualifiers except /STATUS.

Outfile Qualifier

The **/OUTFILE=<file>** qualifier redirects the output of data to a specified report data file. You might use this qualifier when you have a cyclic report execution, for example when you are using a batch file and do not want it to be overwritten.

Normally, report data is put into a file with the same name as the report definition file, plus the extension **.WKD** or **.XLD**. Exceptions to this are reports with the EVLC label, which are put into report data files with the extensions **.001**, **.002**, **.003**, etc.; and reports with TRIGG labels, which create output files with the name defined in the label, plus the extensions **.001**, **.002**, etc. These labels override the /OUTFILE qualifier.

Use the /OUTFILE qualifier as it is used in this example:

```
SV100WIN REPORT01.WK1 /OUTFILE=REP01001.DAT
```

This command creates a report data file named **REP01001.DAT** rather than the default name **REPORT01.WKD**.

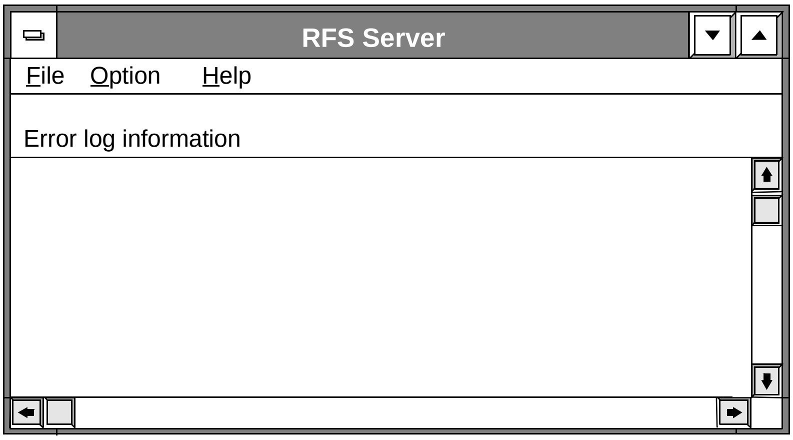
4.2.1.3 The DDE Interface

You can request data from, and send data to, the ABB automation system with the DDE interface. You can also send commands to the SuperView 100 Report software, as you would with the command line interface.

The syntax you use for the DDE interface depends on your DDE application. In this manual, Microsoft Excel is used in the DDE examples, but the DDE syntax for other Windows applications will be similar.

4.2.2 The RFS Server

The following window is displayed when you double-click on the RFS Server icon on the main window:



Error log information

Figure 4-7. The RFS Server

The RFS Server is always executing in a passive mode. When the RFS Server is started, file transfer is always made at the initiative of the remote node (the RFS client).

The File Menu

Use the **File** menu to exit the RFS Server. Choose **File** from the main window menu bar. The following drop-down menu is displayed:

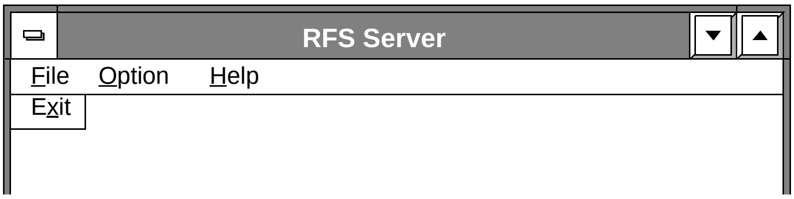


Figure 4-8. The File Menu

- Choose **Exit** to exit the RFS Server program.

The Option Menu

Use the **Option** menu to choose a number of RFS Server software utilities. Choose **Option** from the main window menu bar. The following drop-down menu is displayed:

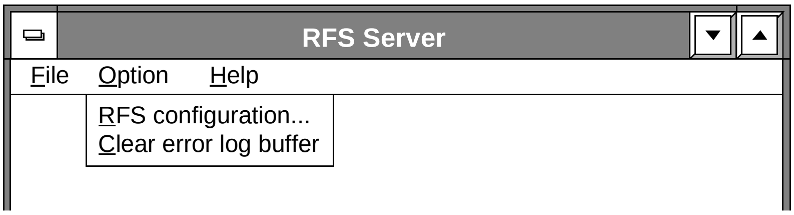


Figure 4-9. The Option Menu

- Choose **RFS configuration** to configure the RFS communication parameters.
- Choose **Clear log buffer** to clear the messages in the log window.

Help Menu

Use the **Help** menu when you need help with, or additional information about, RFS Server software. Choose **Help** from the main window menu bar. The following drop-down menu is displayed:

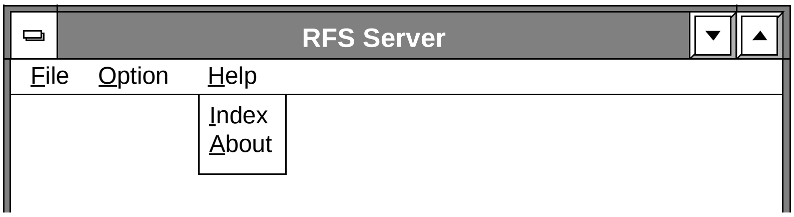


Figure 4-10. The Help Menu

- Choose **Index** to access help information about the RFS Server software.
- Choose **About** to display the current version of the RFS Server software.

4.2.3 The RFS File Transfer Library

The RFS file transfer library is implemented as a **DLL (Dynamic Link Library**) which can be used by user (MS-Windows) applications for file transfer to remote nodes.

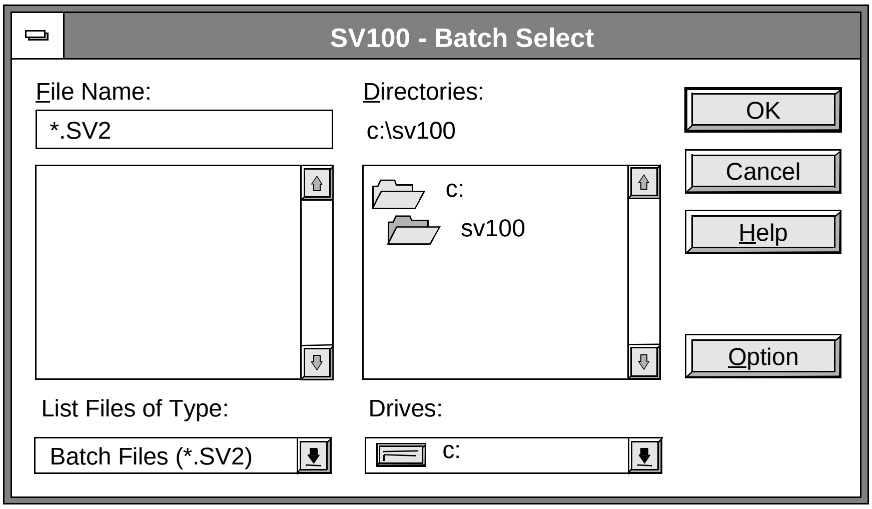
A DLL is dynamically linked during runtime, not linked during the compilation. This gives you the flexibility to use any compiler and linker you choose to create an RFS application.

4.3 Operating Instructions

4.3.1 Batch File Execution

Selecting a Batch File

Choose **Batch select** from the **File** drop-down menu. The following dialog box is displayed:



SV100 - Batch Select

File Name: *.SV2

Directories: c:\sv100

List Files of Type: Batch Files (*.SV2)

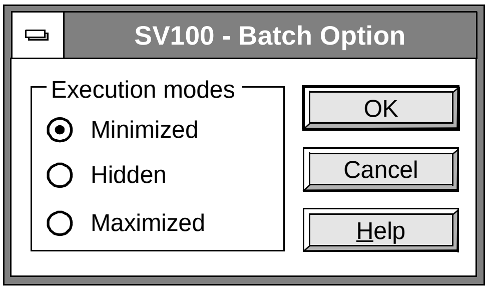
Drives: c:

Figure 4-11. Batch File Selection Dialog Box

If you named your batch files with extension .SV2, they are automatically listed in the list box. Click on the file name for the batch file you want and then click on **OK**. Note, however, that the batch file is just selected, it does not begin to execute. To start the batch execution, use the **Contro**l menu.

Batch Option

You can also set up some extra options for the batch execution when you select the batch file. Click on the **Option** button. The following dialog box is displayed:

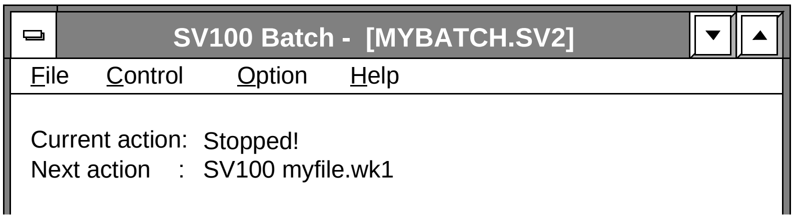


Execution modes
● Minimized
○ Hidden
○ Maximized

Figure 4-12. Option Dialog Box for Batch File Selection

You can control the execution mode of all applications started with the **EXEC** command from the batch file using this dialog box.

After you select a batch file, the caption bar in the main window is updated with the name of the current batch file. In the client area, the first command to execute is displayed as follows:



Current action: Stopped!
Next action : SV100 myfile.wk1

Figure 4-13. Client Area After a Batch File Selection

Batch Execution

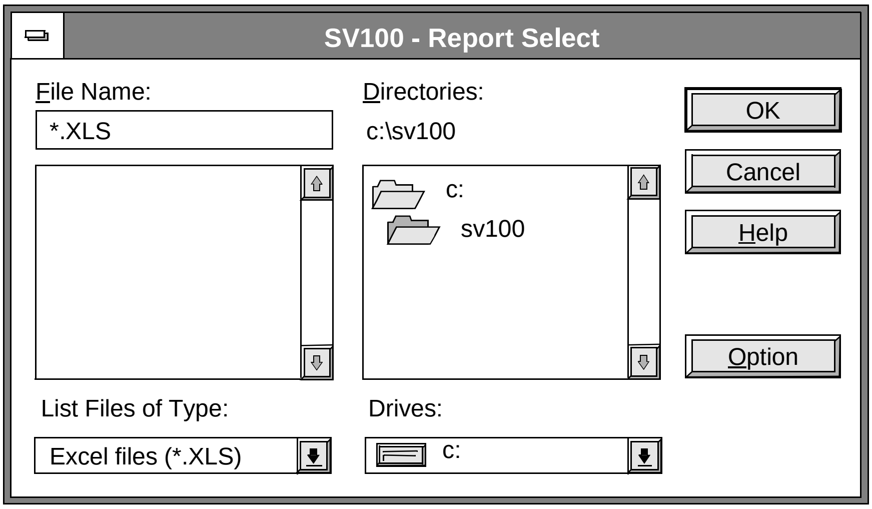
You can control the batch file execution from the **Control** menu. You can start, stop and restart the batch file execution from this menu.

The **Current action** and **Next action** fields, which are displayed in the client area, are continuously updated during the batch file execution.

4.3.2 Report Execution

Selecting a Report

Choose **Report exec** from the **File** drop-down menu to select a report file for interactive execution. The following dialog box is displayed:



File Name: *.XLS

Directories: c:\sv100

List Files of Type: Excel files (*.XLS)

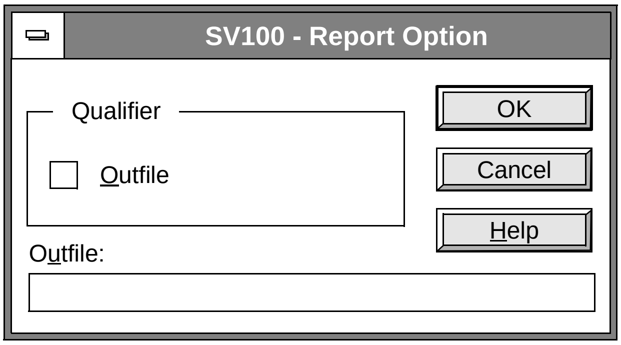
Drives: c:

Figure 4-14. The Report Execution Dialog Box

Reports with the extension .XLS are automatically listed in the list box. Double-click on the report file name or click on the report file name and then the **OK** button to start report execution.

Report Option

To add extra options to the report execution, click on the **Option** button. The following dialog box is displayed:



☐ Outfile

Outfile:

Figure 4-15. The Option Dialog Box for Report Selection

Use the Outfile field to redirect output data from the default output report name to a user-defined file.

Report Execution

During report execution, the caption bar of the main window is updated with the file currently executing as follows:

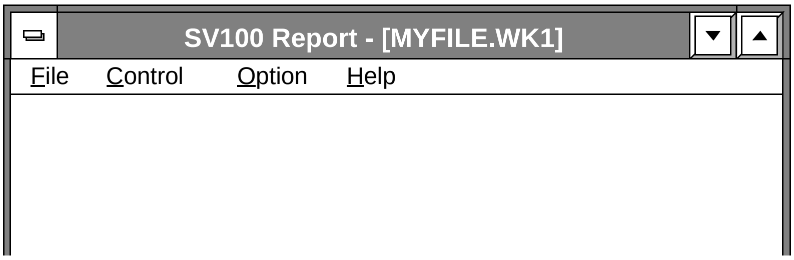


Figure 4-16. Caption Bar when a Report is Executing

Choose **Stop report execution** from the **Control** menu to stop report execution before it finishes. You will know that the report execution is completed when the **SuperView 100** caption returns to the main window’s caption bar.

4.3.3 Symbol Resolution

Choose **Resolve** from the **Option** menu to access symbol resolution.

Symbol resolution translates all symbols in the GCOMSYMB.DTA and TTDSYMB.DTA symbol files. If symbols or TTD symbols are found that cannot be resolved, they are written to symbol error files named GCOMSYMB.ERR or TTDSYMB.ERR, respectively.

During the symbol resolution, no report or batch execution can be made. The main window caption bar is updated as follows,

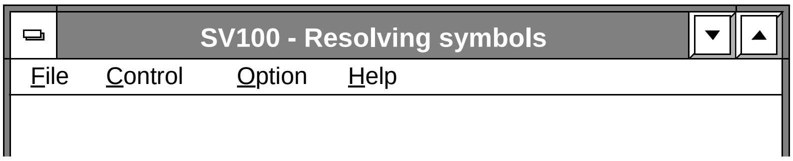

SV100 - Resolving symbols

Figure 4-17. Caption Bar During Symbol Resolution

and then:

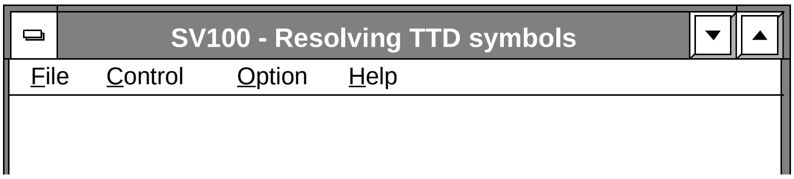

SV100 - Resolving TTD symbols

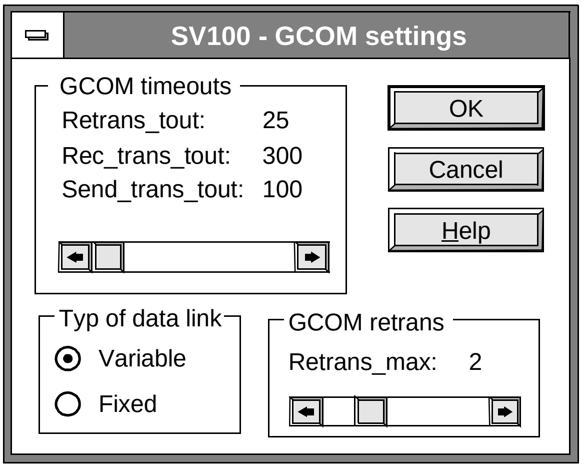
Figure 4-18. Caption Bar During TTD Symbol Resolution

You will know the symbol resolution is completed when the **SuperView 100** caption returns to the main window's caption bar.

4.3.4 GCOM Configuration

You can configure the GCOM device as to time-outs, the number of retransmissions, etc. It is recommended, however, that you always use the default values and never change them before discussing it with ABB Automation.

Choose **GCOM settings** from the **Option** menu. The following dialog box is displayed:



SV100 - GCOM settings

GCOM timeouts
Retrans_tout: 25
Rec_trans_tout: 300
Send_trans_tout: 100

Typ of data link
● Variable
○ Fixed

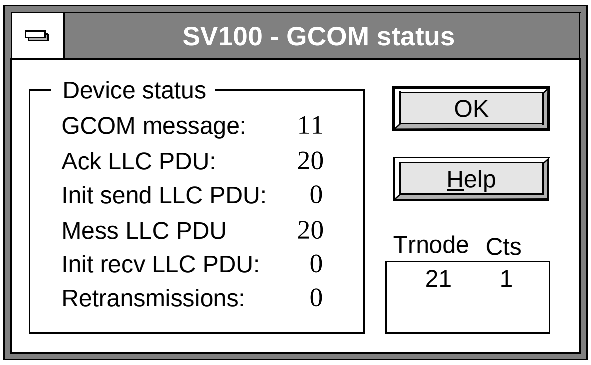
GCOM retrans
Retrans_max: 2

Figure 4-19. GCOM Settings Dialog Box

All the changes you make will be saved in the **GCOMDESC.DTA** file, which is used by the GCOM device for setup information. To activate the changes for the GCOM device, reconnect the GCOM network by first choosing **GCOM disconnect** and then **GCOM connect** from the **Option** menu.

4.3.5 GCOM Status

Choose **Comm. status** from the **Option** menu to determine the current status of the GCOM device. The following dialog box is displayed:



SV100 - GCOM status

Device status

GCOM message: 11
Ack LLC PDU: 20
Init send LLC PDU: 0
Mess LLC PDU 20
Init recv LLC PDU: 0
Retransmissions: 0

Trnode	Cts
21	1

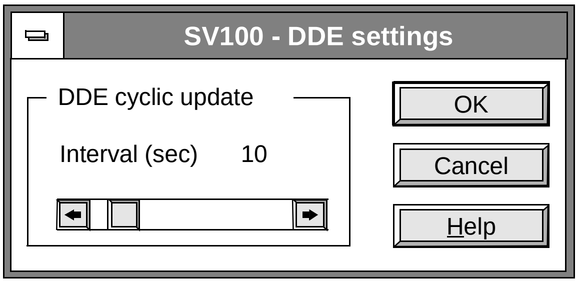
Figure 4-20. GCOM Status Dialog Box

You will find information in this dialog box such as the number of GCOM messages and acknowledge messages sent from the GCOM device.

The list of transit nodes that appears in the list box is the most important information in this dialog box because that list tells you which nodes are directly connected to the GCOM network and have contact with SuperView 100.

4.3.6 DDE Configuration

DDE has a dynamic updating default set for every 10 seconds. To modify this setting, choose **DDE settings** from the **Optio**n menu. The following dialog box is displayed:



SV100 - DDE settings

DDE cyclic update

Interval (sec) 10

Figure 4-21. DDE Settings Dialog Box

You can set the update time between 5 and 60 seconds.

4.3.7 GCOM Network Connection

The GCOM network is automatically connected when the SuperView 100 Report software is started. To disconnect from, or reconnect to, the GCOM network, do the following:

- Choose **GCOM disconnect** from the **Option** menu to disconnect from the GCOM network.
- Choose **GCOM connect** from the **Option** menu to reconnect to the GCOM network.

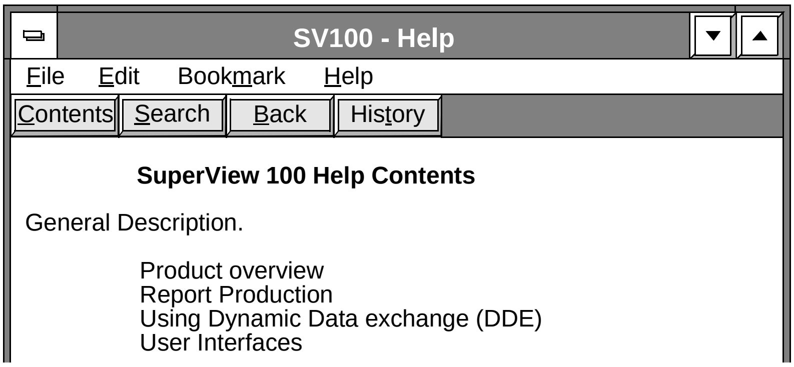
It is not recommended that you disconnect or reconnect often because it will generate system messages in the other nodes in the ABB automation system. The GCOM device should still be connected to the GCOM network when the SuperView 100 Report software is terminated.

4.3.8 Help

You can access the SuperView 100 Report software Help function in the following ways:

- Choose **Help** from the main window menu bar.
- Click on the **Help** button in any dialog box during report execution, batch file selection or configuration.

The Help function gives you on-line help information for all functions of the SuperView 100 Report software. Help information is displayed as follows:



SuperView 100 Help Contents

General Description.

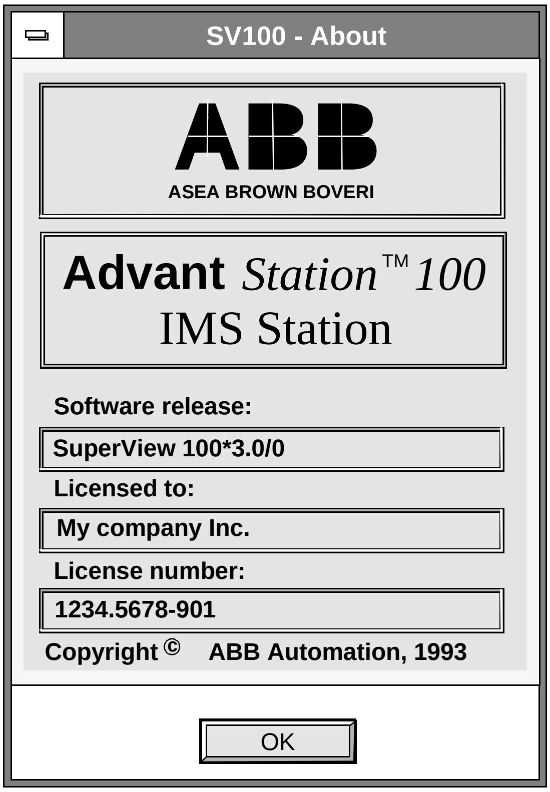
Product overview
Report Production
Using Dynamic Data exchange (DDE)
User Interfaces

Figure 4-22. Help Information

The Help information for the RFS Server is displayed in the same way.

4.3.9 Current SuperView 100 Version

Choose **About** from the **Help** menu to determine what version of SuperView 100 software you currently have. The following dialog box is displayed:



Software release:
SuperView 100*3.0/0

Licensed to:
My company Inc.

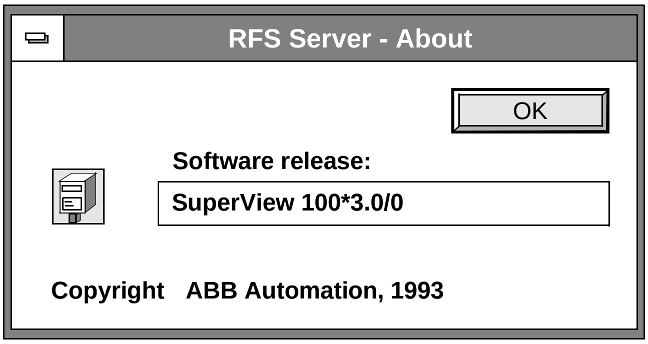
License number:
1234.5678-901

Copyright © ABB Automation, 1993

Figure 4-23. SuperView 100 About Dialog Box

You can also check your SuperView 100 licence number from this dialog box. Always use this licence number when you contact ABB for help.

The current SuperView 100 version can also be obtained from the RFS Server’s **About** function The following dialog box is displayed:



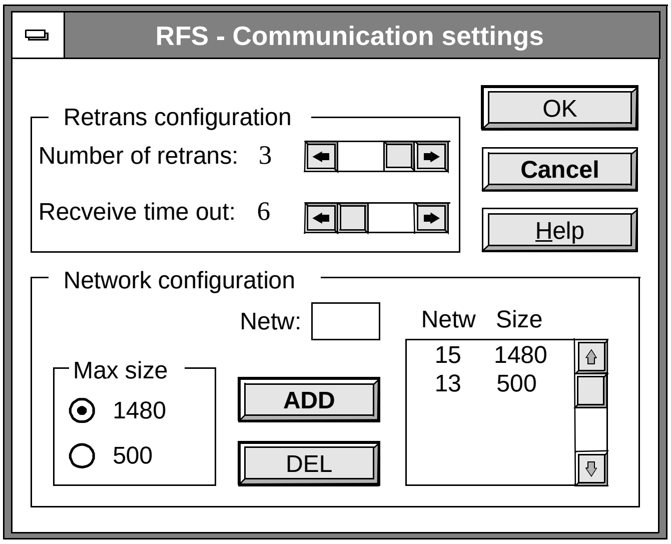
Software release:
SuperView 100*3.0/0

Copyright   ABB Automation, 1993

Figure 4-24. RFS Server About dialog box

4.3.10 RFS Configuration

Choose **RFS configuration** from the **Option** menu. The following dialog box is displayed:



Number of retrans: 3

Recveive time out: 6

Netw:

Max size

● 1480

○ 500

Netw	Size
15	1480
13	500

Figure 4-25. The RFS Configuration Dialog Box

Set up time-outs for the RFS server and define the package size to use for different networks from this dialog box.

NOTE

If a maximum package size of 1,480 bytes is set up for a network, you must verify that your GCOM device is set up for variable package size (by default). Also note that both the Advant Station 500 Information Management Station and MasterBatch are connected on MB 300 networks, and the maximum package size for MB 300 is 500 bytes.

All the changes you make are saved in the RFS configuration file (see Section B.6 RFS Configuration File). To activate these changes, you must restart the RFS server.

Chapter 5 Maintenance

5.1 Preventive Maintenance

As you use the SuperView 100 Report program, the SuperView 100 symbol files grow. This is especially true when data requests are made for a number of different objects.

Many of these objects are probably no longer of use to you. If so, delete the symbol files, and then execute all the reports that you still use. These files will then contain only the object references you currently use. This will decrease the report execution time since the SuperView 100 Report program does not have to search through so many objects during each report's symbol resolution.

5.2 Hardware Indicators

Not applicable.

5.3 Error Messages

Errors are reported in SuperView 100 from three sources:

- Errors detected by the SuperView 100 Report program.
- Errors detected by RFS.
- Errors in report definition labels.

Errors found by the MS-Windows version of the SuperView 100 Report program and the RFS Server are reported in these programs error log window, along with detailed information about the error. See Appendix C MS-DOS Version Specific, for information about error messages for the DOS version of the SuperView 100 Report and RFS software.

Errors in report definitions, unique to the objects in reports, are expressed by **error labels** in the report data file.

5.3.1 Error Labels

Use error labels to indicate errors in report labels or errors that are unique to specific report labels. An error label appears as the first, and only, data item in the data range of the report label involved. The general layout of an error label is:

'#ERR = *<error message>*'

The *<error message>* parameter falls into one of these two major groups.

1. The *<error message>* parameter refers to the sequential number of the erroneous parameter in the report label. The error label has the following general layout:

   ' **#ERR = Error in parameter** *<parameter number>*'

   where the *<parameter number>* can take any value from 1 to 19.

2. The <*error message*> parameter describes an error, local to the report label, which has caused data to fail to appear. The error label can have any of the following layouts:

   #ERR = Insufficient memory'

   There is insufficient memory for a data buffer.

   #ERR = No data'

   There is a lack of data, possibly due to an error in the object specification, no communication or overload. Refer to Section 5.4 Fault-Finding and User Repair, Fault-Finding and User Repair.

   #ERR = Unresolved object'

   The object is not resolved, possibly due to an error in the object name.

   #ERR = Unreachable node'

   There is no communication with the node concerned.

   #ERR = Illegal log interval'

   The TTD log interval of a logged object is illegal.

   #ERR = Illegal property'

   The TTD property of a logged object does not exist.

   #ERR = Unresolved TTD'

   A TTD object is not resolved, possibly due to an error in the object name, the log interval, the log function or the property.

   #ERR = Too many TTD values by row'

   A maximum of 240 TTD values can be presented by row. If there are more, they must be presented by column.

   #ERR = No events in event list'

   There are no events in the event list or the requested time is too old.

   #ERR = DAT configuration mismatch'

   The number of DATs requested are not configurated in the remote node.

5.4 Fault-Finding and User Repair

Communication problems are the most common cause of SuperView 100 software errors. Before any report is executed or any file transferred, verify that SuperView 100 has contact with the rest of the GCOM network, as described in Section 2.5 Product Verification.

Inability to Communicate with GCOM

If you are unable to make contact with the GCOM network (no trnodes are listed), ensure that:

- Other GCOM nodes are up and running.
- The jumper for heartbeat control in the external transceivers is set to *on*.
- The jumper position on the IEEE 802.3 board in the PC is correct for thin (10Base2) and thick (10Base5) Ethernet cable.
- There is no device conflict regarding the IRQ, RAM area and I/O address.

No Data Received with the SuperView 100 Report Software

If the SuperView 100 report software does not receive any data, an **#ERR= No data** error label occurs. One of the following may be the cause of the error:

- **Communication with the node is lost**. Check the communication status.
- **An illegal object reference exists.** The object reference has been changed and is no longer valid in SuperView 100. Resolve the symbol files.
- **There is no MV 800/1 option.** To use MP 200/1, it must be equipped with options for MV 800/1 functionality. Otherwise, only DSC or TSC communication is available with this node and the MP 200/1 can not be used as a transit node for communication with other nodes in the ABB automation system.
- **Overload.** GCOM traffic is so heavy that SuperView 100 is losing messages because the GCOM device’s message queue is full. This should not normally occur.

Binary Files Transferred with RFS are Corrupted

If binary files transferred with RFS are corrupted, ensure that the files were transferred in the binary mode and not as ASCII files, which is the default set by the RFS client application.

Both ASCII and Binary Files Transferred with RFS are Corrupted

If both ASCII and binary files transferred with RFS are corrupted, check if there is a MB 300 network involved in the communication (for example communication between a control network and plant network). If so, ensure that the maximum packet size has been limited to 500.

Appendix A Network and Node Configuration Rules

A.1 General

This section describes the node and network numbers used by SuperView 100. The conventions followed by these node and network numbers differ depending on where SuperView 100 is placed—either in a control network or in the plant network. Examples of each case follow.

Refer to the *ABB MasterNet User's Manual* for complete node configuration rules.

SuperView 100 in a Control Network

In the control network configuration, SuperView 100 is connected to a MasterPiece 200/1 with a DSCS 150 board to implement the GCOM protocol and a DSCS 140 board to handle communication with the control network through MasterBus 300.

Table A-1. Control Network Configuration Rules

Network Number: DSCS 140	A unique number within the control network. 11..19 for control network 1 ... 91..99 for control network 9
Node Number: DSCS 140	A unique number within the control network in the range 1..99
Network Number: DSCS 150	A unique number within the control network. 11..19 for control network 1 ... 91..99 for control network 9
Node Number: DSCS 150	The same number as the one chosen for the DSCS 140 board.
Network Number: SuperView 100	The same number as the one chosen for DSCS 150 board.
Node Number: SuperView 100	A unique number within the control network in the range 1..99

Note that both the DSCS 150 and the DSCS 140/145 boards must be strapped with network and node numbers. See the *ABB MasterGate 230/1 - Installation and Error Codes Manual* for complete instructions on how to strap the DSCS 150 board.

The control network configuration rules are applied to the network in the example below.

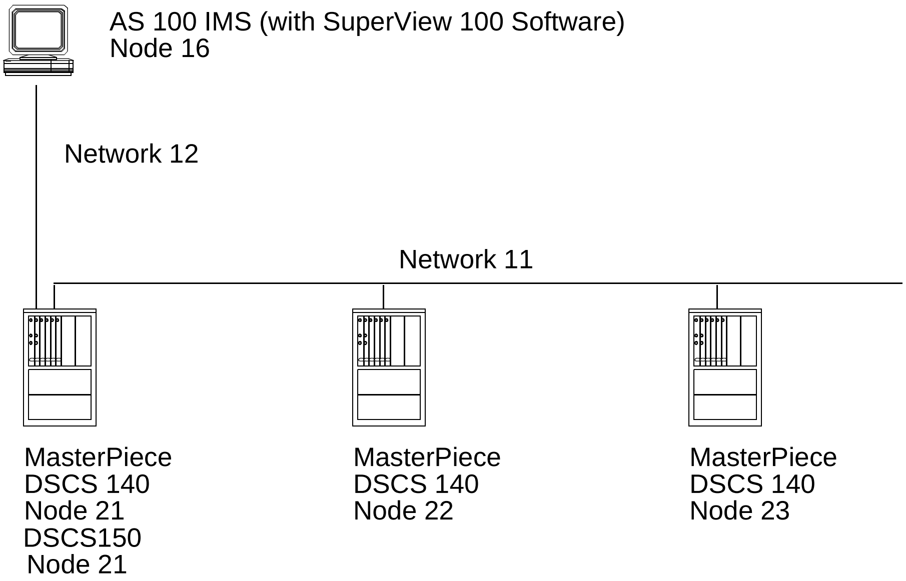


Figure A-1. Example of a Control Network

SuperView 100 on the Plant Network

In the plant network configuration, SuperView 100 is connected to a MasterGate with a DSCS 150 board for the GCOM protocol and a DSCS 140 board for a MasterBus 300 connection to the control network.

Table A-2. Plant Network Configuration Rules

Network Number: DSCS 140	A unique number within the control network. 11..19 for control network 1 ... 91..99 for control network 9
Node Number: DSCS 140	Always 127 in MG 230/1 on control network side
Network Number: DSCS 150	A unique number in the range 110..119.
Node Number: DSCS 150	The number of the control network that the MasterGate is connected to. Node 1 for control network 1, node 2 for control network 2, etc.
Network Number: SuperView 100	The same number as the one chosen for the DSCS 150.
Node Number: SuperView 100	Determined by control network of MG 230/1. Node number 10..19 for control network 1 ... Node number 90..99 for control network 9

The plant network configuration rules are applied to the two networks in the examples below.

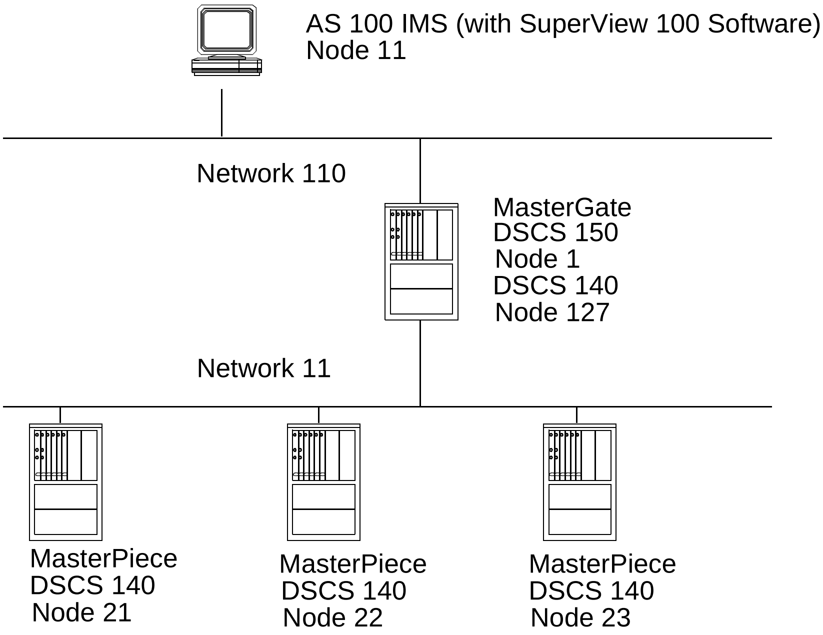


Figure A-2. Plant Network, Example 1

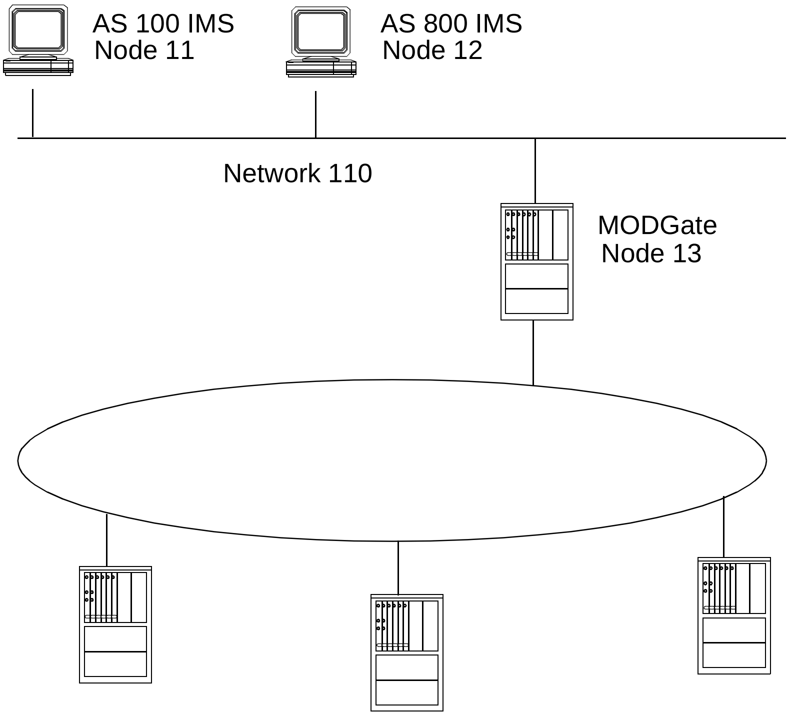


Figure A-3. Plant Network, Example 2

Appendix B Files Created by SuperView 100

B.1 General

Four data files are created by default in the current directory when you activate the SuperView 100 Report software:

- The first data file contains setup information used by SuperView 100 to start up GCOM communication. This file is named **GCOMDESC.DTA**.
- The next two data files contain all the resolved symbolic names of objects and TTD logs in remote control networks which are used by SuperView 100. These files are named **GCOMSYMB.DTA** and **TTDSYMB.DTA**, respectively.
- The last data file contains setup information of the SuperView 100 Report software. This file is only used by the MS-Windows version of SuperView 100 and is named **SV100WIN.INI**.

Set the default directory for these files by assigning a value to the parameter **SVFILES** in the DOS environment. For example, if you are going to put the files into a directory called SV100 on drive C, include the following command in the **AUTOEXEC.BAT** file:

```
SET SVFILES=C:\ 123
```

This setting overrides the default directory used during report requests.

When you use the RFS in SuperView 100, two types of files are created.

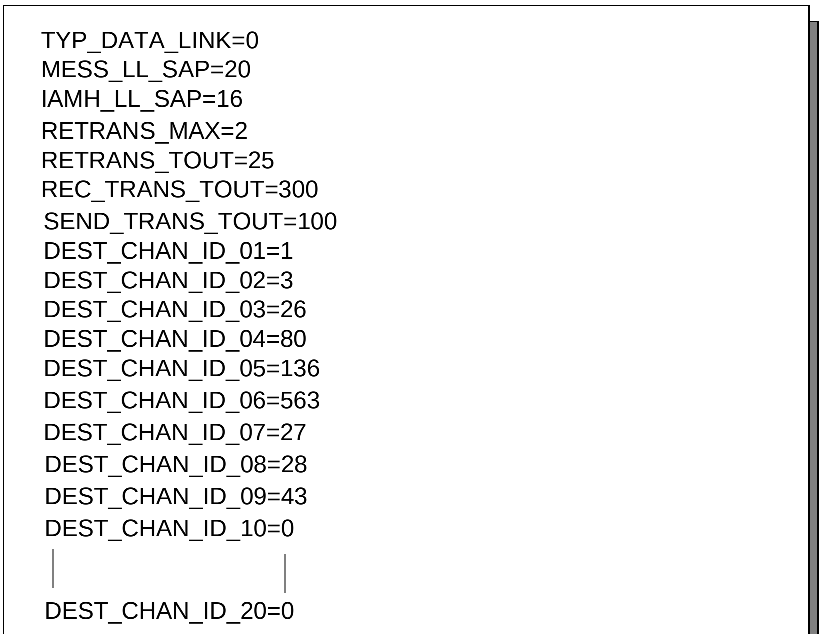
- **Description files** are created for all files created by the RFS. For example, when a data file is fetched from a remote node by the RFS, a description file describing the record structure of the fetched data file is created by the RFS. The description file is given the same name as the data file, plus the extension **.RFS**.
- The **RFS configuration file** is created by the RFS configuration program. You can set up communication parameters for the RFS from this file. The directory and name for this file are set up by assigning a value to the parameter **RFS_CONF** in the DOS environment as follows:

  ```
  SET RFS_CONF=C:\ RFS\ RFSCNFE0.DAT
  ```

B.2 Setup File

The setup file is an ordinary ASCII file named GCOMDESC.DTA which contains setup information about:

- The type of data link
- GCOM addresses
- The maximum number of retransmissions
- Time-out factors
- Supported ABB Master channel identities



```
TYP_DATA_LINK=0
MESS_LL_SAP=20
IAMH_LL_SAP=16
RETRANS_MAX=2
RETRANS_TOUT=25
REC_TRANS_TOUT=300
SEND_TRANS_TOUT=100
DEST_CHAN_ID_01=1
DEST_CHAN_ID_02=3
DEST_CHAN_ID_03=26
DEST_CHAN_ID_04=80
DEST_CHAN_ID_05=136
DEST_CHAN_ID_06=563
DEST_CHAN_ID_07=27
DEST_CHAN_ID_08=28
DEST_CHAN_ID_09=43
DEST_CHAN_ID_10=0
|                 |

DEST_CHAN_ID_20=0
```

Figure B-1. The Contents of the Setup Information File

You can edit the setup file from the MS-Windows version of the SuperView 100 Report software by choosing **GCOM settings** from the **Option** menu. Is is also possible to edit the file in a normal text editor, but it is not advisable for you to change the text editor's default values.

The **TYP_DATA_LINK** parameter selects the packet size for messages on the data link as follows:

- **TYP_DATA_LINK = 0**

  This is the default setting. It selects variable-size packets, 64-1,518 bytes.

- **TYP_DATA_LINK = 1**

  This setting selects fixed-size packets of 510 bytes.

NOTE

The TYP_DATA_LINK parameter must be in accordance with the setup of the GCOM communications board on the ABB automation system side.

B.3 Symbol Files

The symbol files, which contain resolved symbol references, are named **GCOMSYMB.DTA** and **GCOMSYMB.OLD**. GCOMSYMB.DTA is the file used. Whenever translation is requested by the **/RESOLVE** qualifier, GCOMSYMB.DTA is copied to GCOMSYMB.OLD. The files contain the following:

- The names of objects
- The network and node numbers of objects
- The logical file and record numbers of objects
- The reference types of objects

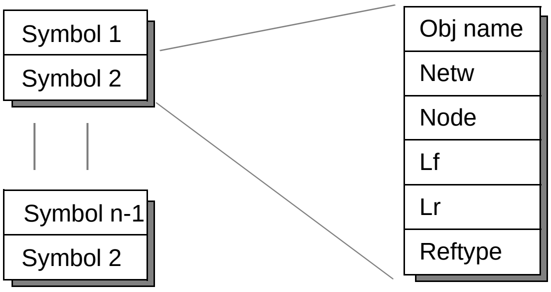


Figure B-2. Symbol File Structure

When the symbols are translated to the SuperView 100 Report program as a result of the /RESOLVE qualifier, a symbol error file containing the remaining, unresolved symbols is created. This error file is named **GCOMSYMB.ERR** and contains ASCII information according to the following figure:

```
<Netw 1> <Node 1> '<Objectname 1>'
<Netw 2> <Node 2> '<Objectname 2>'

<Netw n> <Node n> '<Objectname n>'
```

Figure B-3. Contents of a Symbol Error File

Each line in the symbol error file defines a symbol that remains unresolved. A symbol can remain unresolved for any of the following reasons:

- The object name has been changed.
- The object has been removed.
- There is no contact with the object’s host node.

B.4 TTD Symbol Files

The TTD symbol files, which contain resolved TTD symbol references, are named **TTDSYMB.DTA** and **TTDSYMB.OLD**. TTDSYMB.DTA is the file used. Whenever translation is requested by the /RESOLVE qualifier, TTDSYMB.DTA is copied to TTDSYMB.OLD. The files contain the following:

- The names of objects
- The network and node numbers of objects and TTD logs
- Logical file and record numbers of TTD logs
- Pointers to variables in TTD logs
- Log functions
- Log properties
- Log intervals

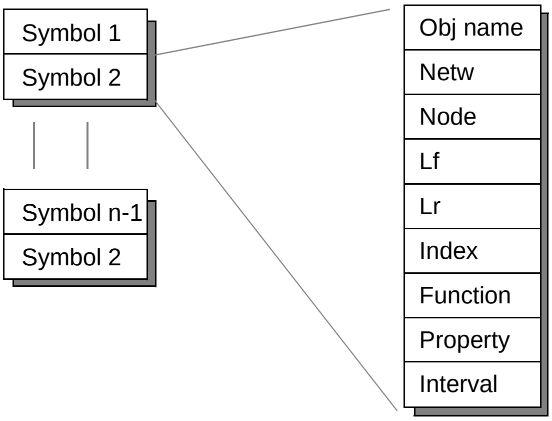


Figure B-4. Structure of a TTD Symbol File

When the symbols are translated to the SuperView 100 Report program as a result of the /RESOLVE qualifier, a symbol error file containing the remaining, unresolved TTD symbols is created. This error file is named **TTDSYMB.ERR** and contains ASCII information according to figure below.

```
<Netw 1> <Node 1> ‘<Objectname 1>’ ‘<Prop>’ ‘<LogInt>’ ‘<Func>’
<Netw 2> <Node 2> ‘<Objectname 2>’ ‘<Prop>’ ‘<LogInt>’ ‘<Func>’
  |                          |

<Netw n> <Node n> ‘<Objectname n>’ ‘<Prop>’ ‘<LogInt>’ ‘<Func>’
```

Figure B-5. Contents of a TTD Symbol Error File

Each line in the TTD symbol error file defines a TTD symbol that remains unresolved. A TTD symbol can remain unresolved for any of the following reasons:

- The TTD log is no longer defined.
- The object name has been changed.
- The object has been removed.
- There is no contact with the object's host node.
- The log has changed regarding property, interval or function.

B.5 RFS Description Files

The RFS description files contain information about the record structure of the corresponding data file as follows:

- Variable or fixed record structure
- Record size
- File extension of the corresponding data file

The description files are used to maintain the data files record structure between different systems.

A description file has the same file name as the file being transferred via the RFS, plus the extension .RFS. If files have the same name, for example, **PROGRAM.C** and **PROGRAM.EXE**, the description file, **PROGRAM.RFS**, contains file information for both files and separates them by file extension.

A description file is always created for files received by an RFS application. Files sent via an RFS application do not necessarily need a description file; in that case, the following default values are used:

- For an ASCII transfer, variable record structure
- For a binary transfer, one byte fixed record structure

If you want a description file, for example, to get a file's special record structure or to increase the transfer rate, use the local file primitives in a user application when creating the file. The **RFSLCD** function is used to create a description file for an already existing file. Refer to Section 6.1.7, RFS Local File Primitives, for more information about local file primitives.

NOTE

The file extension .RFS is reserved by SuperView 100. Files with that extension cannot be transferred by RFS.

B.6 RFS Configuration File

The RFS configuration file contains RFS communication parameters such as:

- Retransmission time-out
- Number of retransmissions
- Maximum bus package sizes

A file transferred by RFS will be split up into a number of packages. The **maximum bus package size** is defined based on the destination node configuration. You can configure up to four different networks with values of 500 or 1,480 bytes (the default value).

If the transferred package is not acknowledged within the **retransmission time-out**, it is sent again. You can set the retransmission time-out to a value between 4 and 60 seconds (the default value is 6 seconds), and the number of retransmissions between 0 and 3 (the default value is 3).

If no RFS configuration file is found, the default values are used.

NOTE

If a MB 200/MB 300 network is used in between the two RFS nodes, the maximum bus packet size is limited to 500.

Appendix C MS-DOS Version Specific

A PC with MS-DOS is a single-user, single-tasking system. You can execute only one program at a time with MS-DOS. For example, if you start the DOS version of the SuperView 100 Report software, you cannot do anything else on the PC while the SuperView 100 Report software is running.

C.1 SuperView 100Report Software

The SuperView 100 Report software takes command-line parameters as qualifiers to define program execution. The general command syntax is as follows:

```
SV100 [drive:][path]filename.ext [qualifiers]
```

The command line parameters are:

- ***drive***: is the optional disk drive specification.
- ***path*** is the optional file path specification.
- ***filename*** is the report definition file name.
- **ext** is the report definition file extension.

The report definition file extensions can be any of the following:

- **WKS** for worksheets produced by Lotus 1-2-3/1.
- **WK1** for worksheets produced by Lotus 1-2-3/2.
- **WR1** for worksheets produced by Lotus Symphony®.
- **XLS** for worksheets produced by Microsoft Excel (versions 2.1, 3.0 and 4.0).

If the SuperView 100 Report software is activated without any parameters or qualifiers, it presents the syntax it expects:

```
SV100: Error 1
Syntax: SV100 [drive:][path]filename.ext [qualifier[qualifier...]]
  Allowed qualifiers:
      /Status             Show status of GCOM subnet
      /Resolve            Resolve all symbols and TTD symbols
      /Disconnect         Disconnect from the GCOM subnet
      /Outfile=<file>     Create the specified report data file
      /Noerror            No error report to the screen
```

Abbreviate the qualifiers by leading character(s), for example, /D or /DIS for disconnect.

Status Qualifier

The **/STATUS** qualifier displays the status of the GCOM subnet on the screen in the following format:

```
GCOM status
gcom_mess_nr = 1   init_rec    = 0   rec_status   = 0

mess_llc_pdu = 3   init_send   = 0   trans_status = 0

ack_llc_pdu  = 3   retarns_nr  = 0

 Transit node   Cts
 ------------------
           21     1
 ------------------
```

Figure C-1. GCOM Subnet Status Display

The transit node table, which lists connected and active nodes in the GCOM subnet, provides the most important information on the status display.

The values following **mess_llc_pdu** and **ack_llc_pdu** should be in the same range. The values following **init_rec**, **init_send** and **retrans_nr** should be low (0). The values following **rec_status** and **trans_status** should be as above.

Use the /STATUS qualifier alone or with other parameters and qualifiers. When you use it in combination with other parameters and qualifiers, the status display is shown first.

Disconnect Qualifier

The **/DISCONNECT** qualifier disconnects SuperView 100 from the GCOM subnet. Use the /DISCONNECT qualifier alone or with other parameters and qualifiers. When you use it with other parameters and qualifiers, /DISCONNECT is executed last.

Resolve Qualifier

The **/RESOLVE** qualifier translates all symbols in the GCOMSYMB.DTA and TTDSYMB.DTA symbol files. If symbols or TTD symbols are found that cannot be resolved, they are written to symbol error files named GCOMSYMB.ERR or TTDSYMB.ERR, respectively.

Use the /RESOLVE qualifier alone or with other parameters and qualifiers. When you use it with other parameters and qualifiers, /RESOLVE is executed before all others, except /STATUS.

See Section B.3 Symbol Files, for further information about symbol files.

Noerror Qualifier

The /NOERROR qualifier suppresses the normal output of error messages to the screen.

You can redirect error messages to a file using the DOS **Redirection** command. For example:

```
SV100 REPORT01.WK1 > REPORT01.ERR
```

This command redirects all output from the SuperView 100 report request program to the **REPORT01.ERR** file.

Outfile Qualifier

The **/OUTFILE=<file>** qualifier redirects the output of data to a specified report data file. Normally, report data is put into a file with the same name as the report definition file, plus the extension **.WKD**. The exceptions to this are:

- Reports with the EVLC label, which are put into report data files with the extensions **.001**, **.002**, **.003**, etc.
- The TRIGG label, which creates output files with the same name as defined in the label plus the extensions **.001**, **.002**., **.003**, etc.

These labels override the /OUTFILE qualifier.

Use the /OUTFILE qualifier as it is used in this example:

```
SV100 REPORT01.WK1 /OUTFILE=REP01001.DAT
```

This command creates a report data file named **REP01001.DAT** instead of the default REPORT01.WKD. For EVLC reports, the extension given is not used. For TRIGG labels, the data file name will be defined by the label itself.

C.2 Cyclic Report Requests

You can request cyclic reports using the **ABSWAIT** and **RELWAIT** batch file extenders which are supplied with SuperView 100. Consider ABSWAIT and RELWAIT DOS extensions and use them as DOS commands in batch files.

The ABSWAIT command takes an absolute time parameter and delays batch file execution until the absolute time specified occurs. The RELWAIT command takes a relative time parameter and delays batch file execution for the relative time specified.

Use the ABSWAIT and RELWAIT commands to control execution of report request commands in DOS batch files.

The syntax of ABSWAIT is as follows:

ABSWAIT <*absolute time***>**

The <*absolute time*> parameter is specified as hour and minute, separated by a colon,

(HH:MM)

where:

- **HH**, the hour, is specified in a 24-hour format by two digits, 00-23.
- **:** (a colon) separates the hour and the minute.
- **MM**, the minute, is specified by two digits, 00-59.

For example, if you want to specify execution of the next command at 3:30 p.m., enter the command as follows:

ABSWAIT 15:30

The syntax of RELWAIT is as follows:

RELWAIT <*relative time***>**

The <*relative time*> parameter is specified as hour and minute, separated by a colon,

(HH:MM)

where:

- **HH**, the hour, is specified in a 24-hour format by two digits, 00-23.
- **:** (a colon) separates the hour and the minute.
- **MM**, the minute, is specified by two digits, 00-59.

For example, if you want to specify execution of the next command for 90 minutes (1.5 hours) later, enter the command as follows:

RELWAIT 01:30

The figure below shows a simple batch file containing ABSWAIT and RELWAIT cyclic report requests.

```
ECHO cyclic 8 hour report requests
DAILY:
  ABSWAIT 08:00
   SV100 PRODREPORT1.WK1 /OUT=PROD08.WKD
  RELWAIT 08:00
  SV100 PRODREPORT1.WK1 /OUT=PROD16.WKD
  RELWAIT 08:00
  SV100 PRODREPORT1.WK1 /OUT=PROD24.WKD
GOTO DAILY
```

Figure C-2. A Simple Batch File for Cyclic Report Requests

- The **ECHO** command at the beginning of the file echoes the text, **cyclic 8 hour report requests**, to the screen.
- The second line defines a label named **DAILY**.
- The first **ABSWAIT** command waits until **08:00** before executing the next command line.
- At **08:00**, the first report request is executed, creating the **PROD08.WKD** data file.
- The first **RELWAIT** command waits for eight hours until 16:00 before executing the next command line.
- At 16:00, the report data file **PROD16.WKD** is created, and at 00:00 (24:00), **PROD24.WKD** is created.
- The **GOTO** command on the last line returns execution to the line below the **DAILY** label, which resumes execution at 08:00 the next day.

NOTE

The batch file continues execution in an endless loop until it is interrupted by a **<CTRL>+<BREAK>** command.

C.3 RFS File Transfer Utility Program

The RFS File Transfer Utility program (**RFSFTU**) exists only in the DOS version of SuperView 100. You can upload, download and replace sequential (ASCII or binary) files via the remote file system using RFSFTU. This program is an RFS client and expects a remote file server at the remote node.

The following remote file-handling functions are available from the remote file transfer utility, using self-explanatory dialogs.

- Download a file to a remote node.
- Replace a file in a remote node.
- Upload a file from a remote node.
- Delete a file in a remote node.
- Rename a file in a remote node.

Run the utility by entering the following command:

C:\RFSFTU <CR>

The following menu is displayed on the screen:

```
              SuperView 100
  RFS File Transfer Utility
  Terminate with <Ctrl> + <Break>
```

Figure C-3. RFS File Transfer Utility Program Menu

```
1 - Download a file to remote node

2 - Replace a file in a remote node

3 - Upload a file from the remote node

4 - Delete a file in the remote node

5 - Rename a file in the remote node

Enter selection: 1
```

From the menu, choose the operation you want to execute. The dialog for downloading is as follows:

```
Remote network number (11..119) : 110

Remote node number (1..127) : 12

Local file path in POSIX-style : /DEV/MYPROG.EXE

Remote file path (default=local file path) : PROG/PROG.EXE*

File format (0=binary, 1=ASCII default=ASCII) : 0

Delay time in msec (default=0) : 0
```

The dialogs for the other operations are similar.

The RFSFTU program uses description files in the same way as the RFS Server. See Section B.5 RFS Description Files, for more information.

* The remote file path must be given from the RFS root directory, which should not be included in the path specification.

NOTE

You can give the RFSFTU program its parameters directly on the DOS command line, which is useful in batch file applications, etc.

For example,

```
C:>RFSFTU 1 110 12 /DEV/MYPROG.EXE PROG/PROG.EXE 0 0
```

downloads the **C:\DEV\MYPROG.EXE** local file to a remote SuperView 100 node (defined by network = 110, node = 12) with the destination **C:\RFS\PROG\PROG.EXE** as a binary file with 0 ms delay time. (Note that the remote file path is specified without the RFS root directory, which is assumed to be **C:\RFS**).

C.4 RFS Configuration Program

Use the RFS configuration program (**RFSCNF**) in the SuperView 100 DOS version to set up different communication parameters, such as time-out period, number of retransmissions and maximum packet size on the bus. You can make the same configuration settings from a dialog box in the MS-Windows version of the RFS server.

The following parameters can be inspected/defined by means of the remote file system configuration utility, using self-explanatory dialogs.

- Time-out period
- Number of retransmissions
- Maximum packet size on the bus

Enter the following command to run the utility:

C:\RFSCNF <CR>

The following menu is displayed:

```
        SuperView 100
  RFS Configuration program
```

Figure C-4. RFS Configuration Program Menu

```
At all levels or questions, hit <CR> to quit

1 - Inspect parameters

2 - Modify parameters

Selection (1-2) : 2

The menu for modification of parameters is as follows:

Timeout period (4..60, default 6)   : 6 sec.
Number of retransmissions (0..3, default 3)   : 3

Maximum bus packet size (default is max. Ethernet packet
size)

< 1> Network number   : 110
     Max. packet size (500, 1480 default 1480): 1480 Bytes

< 2> Network number   :  15
     Max. packet size (500, 1480 default 1480): 500 Bytes
                    .
                    .
```

Choose the operation you want to execute from the menu. The menu for inspection of parameters is as follows:

```
Timeout period     : 6 sec.
Number of retransmissions     : 3

< 1> Network number: 110  Max. packet size: 1480 Bytes
< 2> Network number:  15  Max. packet size:500 Bytes
                      .
                      .
```

NOTE

If a MB 200/MB 300 network is used in between the two RFS nodes, the maximum bus packet size is limited to 500.

C.5 Error Messages Detected by the SuperView 100 Report Program

The most common error messages detected by the SuperView 100 Report program refer to error situations that you can correct yourself. These error messages are presented as a direct result of program execution. The following list describes possible error messages and the appropriate actions:

Code	Error message and appropriate action
2	**’Can’t access/open input file’** The input file does not exist. Re-issue the command with a correct file name.
3	**’Parameter is not a legal qualifier’** An illegal qualifier was given. Re-issue the command with a legal qualifier.
4	**’Input file is not a worksheet file’** The input file is not a WK1 or XLS format file. Re-issue the command with a legal WK1 or XLS file name.
5	**’Input file contains no report definitions’** No legal report labels were found in the input file. Check the report definition file and reissue the command.
6	**’Can’t access/open output file’** File access error probably due to error in file path specification, insufficient disk memory or hardware error. Check the file path and disk memory and re-issue the command.
110, 202	**’Insufficient memory resources’** Not enough primary memory available for this report. Split the report definition into smaller parts. If in a Lotus session, give the command EXIT and terminate the session.
101, 102	**’Netw or node not available in environment’** The DOS environment variables NETW=, and/or NODE= are missing. Use the DOS ’SET’ command to define or include the commands in the AUTOEXEC.BAT file and restart.
103, 104	**’Netw or node in environment is illegal’** The DOS environment variables NETW= and/or NODE= are illegal. Check and correct the environment variables with the DOS ‘SET’ command, include the corrected commands in the AUTOEXEC.BAT file and restart.
107	**’No communication with transit node(s)’** No communication with the nodes in the GCOM subnet. Use the /STATUS qualifier to verify if communication with the transit nodes is established.
119	**’SV100_ID in environment is illegal’** The DOS environment variables SV100_ID= are illegal. Check and correct the environment variables with the DOS 'SET' command, include the corrected commands in the AUTOEXEC.BAT file and restart.

C.6 Error Messages from the RFS Server

Most of the error messages for the RFS server have a standardized format, consisting of an error text, a log number and a string argument. The error text is a translation of the error code to a DOS system message which is truncated to 40 characters. All these errors are self-explanatory. When they are not self-explanatory, an argument list is supplied to provide more information about each error.

Code	Error message and appropriate action
1	**’MALLOC failure’** An error occurred when RFSSRV attempted to allocate memory during initialization. Check the amount of memory available in the PC.
2	**’RFSSIN failure’** An error occurred when RFSSRV attempted to **initialize. Restart.**
3	**’Failed to receive message’** An error occurred when trying to receive a message. Check the communication.
5,6	**’Failed to initialize the communication board’** Failed to initialize the GCOM communication board. Check the board strappings.
7	**’Failed to send message.** An error occurred when trying to send a message. Check the communication.
12	**’RFSSRV aborting’** RFSSRV has aborted. Check other log messages.
14	**’Failed to read default RFS root directory from the DOS environment’** Failed to read the default RFS root directory from the DOS environment. Check if SV_RFSDIR is defined in the DOS environment.
15	**’Failed to get communication status from transit node’** Failed to get communication status. Check if defined transit node is running.
101	**’Invalid func code’** RFSSRV received a message with an illegal function code. Report to ABB.

C.7 Error Messages from the RFSFTU Program

An error message from the RFSFTU program always contains a standard error text from the run-time library. In some cases, it is followed by a second message with additional error information.

Error messages

’Error opening the GCOM device’
The GCOM device could not be opened by this application. Check the GCOM device and the error code.

’Error connecting the GCOM device’
The GCOM device could not be connected by this application. Check the GCOM device and the error code.

’Error closing the GCOM device’
The GCOM device could not be closed by the application. Check the GCOM device and the error code.

’Error in communication status handling’
The communication status-handling failed. Check the error code and restart the application.

’No communication with remote node’
The remote node is not reachable.

’No communication with transit node(s)’
One or more defined transit nodes are not reachable.

’Error sending signals’
The signal transmission failed, probably due to overload. Check the error code and restart the application.

’Error receiving signals’
Failed to receive signal. Check the error code.

’Time out receiving signals’
No signals received within the specified time.

’Error creating a local file’
Failed to create a file in the local file system. Check the path name.

’Error opening the local file’
Failed to open a file in the local file system. Check the file/path name and the file protection.

’Error getting local file attributes’
Failed to get file attributes for a file in the local file system.

’Error reading from the local file’
Failed to read from a file in the local file system.

’Error writing in the local file’
Failed to write in a file in the local file system.

’Error closing the local file’
Failed to close a file in the local file system.

’Error creating a remote file’
Failed to create a file in the remote file system. Check the path name.

’Error opening the remote file’
Failed to open a file in the remote file system. Check the file/path name and the file protection.

’Error reading from the remote file’
Failed to read from a file in the remote file system.

’Error writing in the remote file’
Failed to write in a file in the remote file system.

’Error closing the remote file’
Failed to close a file in the remote file system.

’Error deleting the remote file’
Failed to delete a file in the remote file system.

’Error renaming the remote file’
Failed to rename a file in the remote file system.

’Error sending translate request to the remote node’
Failed to send a report translate request to a remote node.

’Translate request failed’
Failed to translate a logical report name to physical report name.

’Unknown error’
Error unknown. Check the error code and report to ABB Automation.

Some of these error messages contain an error code at the end of the message. For those error codes that are not described in this chapter, please refer to ABB for further information.

Appendix D Report Labels and Data Ranges

D.1 General

Report labels define objects and their mode of subscription as well as the location and mode of presentation of the collected data. The following table lists the supported report labels.

Table D-1. Supported Labels in Spreadsheet Reports

Label	Defines collection of
#DATE	The date of the report request
#TIME	The time of the report request
#DSC	Data Set (both receipt and dispatch)
#TSC	Text Set (both receipt and dispatch)
#AI	Single AI data
#AO	Single AO data
#DI	Single DI data
#DO	Single DO data
#DAT	Single DAT data
#MDAT	Set of consecutive DAT instances
#TTD	Historical, Time-Tagged data (by log index)
#TTDO	Historical, Time-Tagged data (by object)
#COM	Communication status
#EVL	Process event
#EVLC	Process event (cyclic)
#TRIGG	Trigger data (DSC or TSC)

Each type of report label has its own set of parameters and entry syntax. This syntax and the layout of the corresponding data range are described in detail in the following sections.

The data ranges hold the data collected for the objects of the report which is requested in the report labels. For each type of report label, the layout of data is fixed.

The description of the report labels includes examples. The corresponding data range layouts are described with cell A1 as anchor cell and with the data organized by column (all data in the same column).

D.2 DATE Label and Data Range

Purpose

The **DATE** label puts a date stamp for the date of report generation in the report data file. The date is collected when the report data file is written and is in the legible internal format of the spreadsheet software.

Synopsis

#DATE,<*cell*>[,<*dir*>]

Description

- **<*cell*>** is the address of the data cell, for example, B12 or REL for Relative. REL or Relative means *relative to the definition label.*
- **<*dir*>** is the direction of the data range, that is, C for *by column* or R for *by row*. Direction is only required if the REL cell address is used.

Examples

For example, define the label as follows to obtain a date stamp in cell G3 of a report data file:

#DATE G3

A date stamp is required in the cell directly below the one holding the DATE definition label, which means that all data is organized by column. Define the label as follows:

#DATE REL,C

A date stamp is required in the cell directly to the right of the DATE definition label, which means that all data is organized by row. Define the label as follows:

#DATE REL,R

NOTE

The DATE label can only be used once in each report definition.

Return Value

The data range returned from a DATE label is as follows:

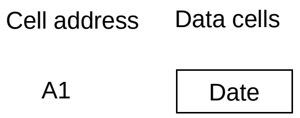


Figure D-1. Data Range Returned from a DAT Label

The data cell containing the date uses the spreadsheet internal data representation for ease of use. By default it is formatted to be legible. The data cell is a floating-point number cell.

Error-Handling

An incorrect parameter in the label is reported by an error message at the position of the data.

D.3 TIME Label and Data Range

Purpose

The **TIME** label puts a time stamp for the time of report generation in the report data file. This time is collected when the report data file is written. The time data is in ASCII form or in the internal format of the spreadsheet software.

Synopsis

#TIME,<*cell*>*[,<dir>,[<pres>]]*

Description

- **<*cell*>** is the address of the data cell, for example, AA27 or REL for Relative. REL or Relative means *relative to the definition label*.
- **<*dir*>** is the direction of the data range, that is, C for *by column* or R for *by row*. Direction is only required if you use the REL cell address or the presentation option.
- **<*pres*>** is the presentation of time, that is, A for ASCII and I (the default) for the internal representation.

Examples

For example, if you want a time stamp in the internal representation in cell H3 of a report data file, define the label as follows:

#TIME H3

A time stamp in ASCII is required in the cell directly below the one holding the TIME definition label, which means that the data is organized by column. Define the label as follows:

#TIME REL,C,A

In the internal representation of the spreadsheet, a time stamp is required in the cell directly to the right of the TIME definition label, which means that all the data is organized by row. Define the label as follows:

#TIME REL,R,I

NOTE

Use the TIME label only once in each report definition.

Return Value

The data range returned from a TIME label is as follows:

Cell address	Data cells
A1	Time

Figure D-2. Data Range Returned from a TIME Label

The data cell containing the time can use the internal format or ASCII for ease of use. The data cell is a floating-point number cell or a label cell.

Error-Handling

An incorrect parameter in the label is reported by an error message at the position of the data.

D.4 DSC Label and Data Range

Purpose

DSC labels collect Data Set (DS) data from the ABB automation system for the report data file or to dispatch DS data to the ABB automation system. Insert data for dispatch into the report definition file.

Synopsis

#DSC,<*cell*>,<*dir*>,<*netw*>,<*node*>,<*ds*>,<*send*>,<*nb*>,<*ni*>,<*nl*>,<*nr*>

Description

- **<*cell*>** is the address of the data cell, for example, AB12 or REL for Relative. REL or Relative means *relative to the definition label*.
- **<*dir*>** is the direction of data, that is, C for *by column* or R for *by row*.
- **<*netw*>** is the network number of the source data.
- **<*node*>** is the node number of the source data.
- **<*ds*>** is the identity number of the source DS.
- **<*send*>**is the direction of transmission, that is, S for *send* or R for *receive*.
- **<*nb*>** is the number of Boolean data items, 0-24.
- **<*ni*>** is the number of integer data items, 0-24.
- **<*nl*>** is the number of long integer data items, 0-24.
- **<*nr*>** is the number of real data items, 0-24.

NOTE

Two sending DSC labels **can not** share the same data cells.

The sum of <*nb*>, <*ni*>, <*nl*> and <*nr*> must not exceed 24, the maximum number of DATs in a single DSC request.

Examples

If you want to receive data from a DS in network 13, node 6, with DS identity = 14, containing 20 DATs grouped as four Booleans, eight integers and eight real numbers, and to organize the data by row from cell EB1 (cells EB1, EC1, ED1, EF1, etc.), define the label as follows:

#DSC,EB1,R,13,6,14,R,4,8,0,8

To send DS data to a DS in node 12, network 23, with DS identity = 7, from cell C5 by column, covering two Booleans, four integers, four long integers and 12 real numbers, define the label as follows:

#DSC,C5,C,23,12,7,S,2,4,4,12

NOTE

Cells C5 and C6 must contain the two Boolean values, cells C7 to C10 the four integers, cells C11 to C14 the four long integer values and cells C15 to C26 the 12 real values.

Return Value

The data range returned from a DSC label is as follows:

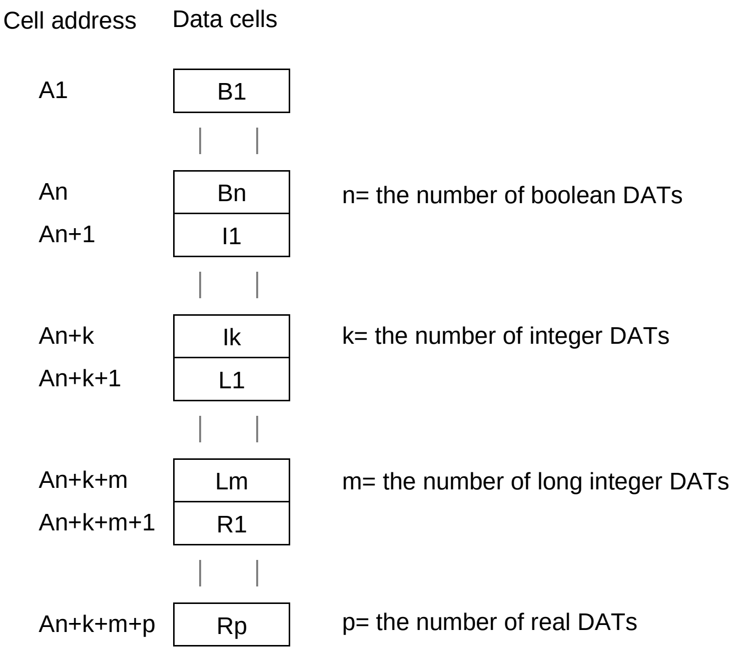

Cell address
Data cells
A1
B1
An
Bn
An+1
I1
n= the number of boolean DATs
An+k
Ik
An+k+1
L1
k= the number of integer DATs
An+k+m
Lm
An+k+m+1
R1
m= the number of long integer DATs
An+k+m+p
Rp
p= the number of real DATs

Figure D-3. Data Range Returned from a DS C Label

In the diagram above, **b**, **i**, **l** and **r** can each can take values 0- 24, but the sum of b+i+l+r cannot exceed 24. The data cells are all floating point number cells. The DSC data range has the same layout for both sending and receiving DSC report entries. Sending data ranges may contain integer, floating point number or formula cells.

Error-Handling

Incorrect label parameters are reported by error messages at the position of the data. Incorrect data dispatch definitions are reported by error messages.

D.5 TSC Label and Data Range

Purpose

TSC labels collect Text Set (TS) data from the ABB automation system for report data files and to dispatch TS data to the ABB automation system. Create data for dispatch in report definition files.

Synopsis

#TSC,<*cell*>,<*dir*>,<*netw*>,<*node*>,<*ts*>,<*send*>

Description

- **<*cell*>** is the address of the data cell, for example, AB12 or REL for Relative. REL or Relative means *relative to the definition label*.
- **<*dir*>** is the direction of the data range, that is, C for *by column* or R for *by row*.
- **<*netw*>** is the network number of the source TS.
- **<*node*>** is the node number of the source TS.
- **<*ts*>** is the identity number of the source TS
- **<*send*>** is the direction of transmission, that is, S for *send* or R for *receive*.

NOTE

Two sending TSC labels **can not** share the same data cells.

The data is always oriented as one long integer, one real value, one Boolean and one text string. The maximum length of the text string is 24, 48 or 72, depending on the definition of the TS in the source node.

Examples

If you want to receive data from a TS in network 13, node 4, with TS identity = 12, into cell FB1 by row (cells FB1, FC1, FD1, FE1, etc.), define the label as follows:

#TSC,FB1,R,13,4,12,R

If you want to send TS data to a TS in network 22, node 8, with TS identity = 4, from cell A10 by column, define the label as follows:

#TSC,A10,C,22,8,4,S

NOTE

Cell A10 must contain the long integer, cell A11 the real value, cell A12 the Boolean value and cell A13 the text string.

Return Value

The data range returned from a TSC label is as follows,

Cell address	Data cells	
A1	Intl	Long integer value
A2	Real	Real value
A3	Bool	Boolean value
A4	String	Text string

Figure D-4. Data Range Returned from a TSC Label

where:

- **Intl** is a floating-point number cell.
- **Real** is a floating-point number cell.
- **String** is a label cell.
- **Bool** is a floating-point number cell defined as a bitmap, each bit holding different information for presentation in the operator station (OS).

Bit 0 = VALID2
Bit 1 = SELECTED
Bit 2 = MAN
Bit 3 = USER_B
Bit 4 = COLOUR1
Bit 5 = COLOUR2
Bit 6 = BLANKT
Bit 7 = BLANKB
Bit 8 = BLANKR
Bit 9 = BLANKIL
Bit 10 = B1_VAL
Bit 11 = B2_VAL
Bit 12 = B3_VAL
Bit 13 = B4_VAL
Bit 14 = `not used`
Bit 15 = `not used`

Error-Handling

An incorrect parameter in the label is reported by an error message at the position of the data. An incorrect data dispatch definition is also reported by an error message.

D.6 AI Label and Data Range

Purpose

AI labels collect analog input data from the ABB automation system for report data files.

Synopsis

#AI,<cell>,<dir>,<netw>,<node>,<name>

Description

- <***cell***> is the address of the data cell, for example, C14 or REL for Relative.
- <***dir***> is the direction of the data range, that is, C for *by column* or R for *by row*.
- <***netw***> is the network number of the source AI object.
- <***node***> is the node number of the source AI object.
- <***name***> is the name of the source AI object.

Example

If you want to obtain data from an AI named TANK2.MV in node 14, network 22, by column from cell C14 (cells C14, C15, C16, etc.), define the label as follows:

#AI,C14,C,22,14,TANK2.MV

Return Value

The data range returned from an AI label is as follows,

Cell address	Data cells	
A1	Status	Status of the request
A2	Rmin	Min. range of the AI value
A3	Rmax	Max. range of the AI value
A4	Unit	Unit of the AI
A5	Value	Value of the AI

Figure D-5. Data Range Returned from an AI Label

where:

- **Rmin** is a floating-point number cell.
- **Rmax** is a floating-point number cell.
- **Unit** is a label cell.

- **Value** is a floating-point number cell.
- **Status** is a floating-point number cell defined as a bitmap where each bit holds different status information.

Bit 0 = IMPLEMENTED
Bit 1 = ERROR
Bit 2 = UPDATED
Bit 3 = UPD_BLK
Bit 4 = MAN_ENTRY
Bit 5 = SELECTED
Bit 6 = NORMAL_OBJ_TREAT
Bit 7 = H2_REP_FAIL_CTRL
Bit 8 = ABOVE_HI_LIM2
Bit 9 = ABOVE_HI_LIM1
Bit 10 = BELOW_LO_H1
Bit 11 = BELOW_LO_H2
Bit 12 = ALARM_UNACK
Bit 13 = REPEAT_FAIL_BLK
Bit 14 = DISTURBANCE
Bit 15 = OVERFLOW
Bit 16 = ALARM_BLK
Bit 17 = ALARM_PERIOD_BLK
Bit 18 = PRINT_BLK
Bit 19 = REPEAT_FAIL_CTRL
Bit 20 = SYNC
Bit 21 = SYNC_UPD
Bit 22 = NOERR_AT OVF
Bit 23 = TESTED
Bit 24 = ACC_ERR
Bit 25 = ACC_LOCK
Bit 26 = L2_REP_FAIL_CTRL
Bit 27 = ER_REP_FAIL_CTRL
Bit 28 = REPEAT_FAIL_CTRL
Bit 29 = AI_ACTION
Bit 30 = AI_RETRY
Bit 31 = ERR_CTRL

Error-Handling

An incorrect label parameter is reported by an error message at the position of the data.

D.7 AO Label and Data Range

Purpose

AO labels collect analog output data from the ABB automation system for report data files.

Synopsis

#AO,<cell>,<dir>,<netw>,<node>,<name>

Description

- ***<cell>*** is the address of the data cell, for example, D12 or REL for Relative.
- ***<dir>*** is the direction of the data range, that is, C for *by column* or R for *by row*.
- ***<netw>*** is the network number of the source AO object.
- ***<node>*** is the node number of the source AO object.
- ***<name>*** is the name of the source AO object.

Example

If you want to obtain data from an AO named PUMP1.SP in node 12, network 20, by row from A10 (cells A10, A11, etc.), define the label as follows:

#AO,A10,R,20,12,PUMP1.SP

Return Value

The data range returned from an AO label is as follows,

Cell address	Data cells	
A1	Status	Status of the request
A2	Rmin	Min. range of the AO value
A3	Rmax	Max. range of the AO value
A4	Unit	Unit of the AO
A5	Value	Value of the AO

Figure D-6. Data Range Returned from an AO Label

where:

- **Rmin** is a floating-point number cell.
- **Rmax** is a floating-point number cell.
- **Unit** is a label cell.

- **Value** is a floating-point number cell.
- **Status** is a floating-point number cell, defined as a bitmap, where each bit holds different status information.

Bit 0 = IMPLEMENTED
Bit 1 = ERROR
Bit 2 = UPDATED
Bit 3 = OUTP_BLK
Bit 4 = not used
Bit 5 = SELECTED
Bit 6 = NORMAL_OBJ_TREAT
Bit 7 = MAN_MODE
Bit 8 = ON_MAX_LIM
Bit 9 = not used
Bit 10 = not used
Bit 11 = ON_MIN_LIM
Bit 12 = ALARM_UNACK
Bit 13 = not used
Bit 14 = not used
Bit 15 = not used
Bit 16 = ALARM_BLK
Bit 17 = ALARM_PERIOD_BLK
Bit 18 = PRINT_BLK
Bit 19 = OUTP_RESTART
Bit 20 = LOCK
Bit 21 = USE_MAX_LIM
Bit 22 = USE_MIN_LIM
Bit 23 = TESTED
Bit 24 = not used
Bit 25 = not used
Bit 26 = LINKED
Bit 27 = not used
Bit 28 = not used
Bit 29 = not used
Bit 30 = not used
Bit 31 = not used

Error-Handling

An incorrect label parameter is reported by an error message at the position of the data.

D.8 DI Label and Data Range

Purpose

DI labels collect digital input data from the ABB automation system for report data files.

Synopsis

#DI,<cell>,<dir>,<netw>,<node>,<name>

Description

- ***<cell>*** is the address of the data cell, for example, A24 or REL for Relative.
- ***<dir>*** is the direction of the data range, that is, C for *by column* or R for *by row*.
- ***<netw>*** is the network number of the source DI object.
- ***<node>*** is the node number of the source DI object.
- ***<name>*** is the name of the DI object.

Example

If you want to obtain data from a DI named FAN3.SW in node 14, network 22, by row directly below the label cell, define the label as follows:

#DI,REL,R,22,14,FAN3.SW

Return Value

The data range returned from a DI label is as follows,

Cell address	Data cells	
A1	Status	Status of the request
A2	Value	Value of the DI

Figure D-7. Data Range Returned from a DI Label

where:

- **Value** is an integer cell.
- **Status** is a floating-point number cell defined as a bitmap where each bit holds different status information.

  Bit 0 = IMPLEMENTED
  Bit 1 = ERROR
  Bit 2 = UPDATED
  Bit 3 = UPD_BLK
  Bit 4 = MAN_ENTRY

Bit 5 = SELECTED
Bit 6 = NORMAL_OBJ_TREAT
Bit 7 = ERR_CTRL
Bit 8 = VALUE
Bit 9 = SEC_VALUE
Bit 10 = NORM_POSN
Bit 11 = SEL_NORM_POSN
Bit 12 = ALARM_UNACK
Bit 13 = REPEAT_FAIL_BLK
Bit 14 = DISTURBANCE
Bit 15 = CALC_VALUE
Bit 16 = ALARM_BLK
Bit 17 = ALARM_PERIOD_BLK
Bit 18 = PRINT_BLK
Bit 19 = REPEAT_FAIL_CTRL
Bit 20 = LOCK
Bit 21 = INVERTED
Bit 22 = DUAL_IND
Bit 23 = TESTED
Bit 24 = `not used`
Bit 25 = V9_CONTROL
Bit 26 = LINKED
Bit 27 = DI_RETRY
Bit 28 = DIC_IND_RED
Bit 29 = DI_ACTION
Bit 30 = DIC_IND_YELLOW
Bit 31 = DIC_IND_GREEN

Error-Handling

An incorrect label parameter is reported by an error message at the position of the data.

D.9 DO Label and Data Range

Purpose

DO labels collect digital output data from the ABB automation system for report data files.

Synopsis

#DO,<cell>,<dir>,<netw>,<node>,<name>

Description

- ***<cell>*** is the address of the data cell, for example, G19 or REL for Relative.
- ***<dir>*** is the direction of the data range, that is, C for *by column* or R for *by row*.
- ***<netw>*** is the network number of the source DO object.
- ***<node>*** is the node number of the source DO object.
- ***<name>*** is the name of the source DO object.

Example

If you want to obtain data from a DO named VALVE2.POS in node 16, network 11, by column from B2 (cells B2, B3, B4, etc.), define the label as follows:

#DO,B2,C,11,16,VALVE2.POS

Return Value

The data range returned from a DO label is as follows,

Cell address	Data cells	
A1	Status	Status of the request
A2	Value	Value of the DO

Figure D-8. Data Range Returned from a DO Label

where:

- **Value** is an integer cell.
- **Status** is a floating-point number cell defined as a bitmap where each bit holds status information:

  Bit 0 = IMPLEMENTED
  Bit 1 = ERROR
  Bit 2 = UPDATED
  Bit 3 = OUTP_BLK
  Bit 4 = not used

Bit 5 = SELECTED
Bit 6 = NORMAL_OBJ_TREAT
Bit 7 = MAN_MODE
Bit 8 = VALUE
Bit 9 = `not used`
Bit 10 = START_VALUE
Bit 11 = `not used`
Bit 12 = ALARM_UNACK
Bit 13 = `not used`
Bit 14 = `not used`
Bit 15 = `not used`
Bit 16 = ALARM_BLK
Bit 17 = ALARM_PERIOD_BLK
Bit 18 = PRINT_BLK
Bit 19 = OUTP_RESTART
Bit 20 = LOCK
Bit 21 = INVERTED
Bit 22 = `not used`
Bit 23 = TESTED
Bit 24 = ORDER_TO
Bit 25 = ORDER_FROM
Bit 26 = LINKED
Bit 27 = `not used`
Bit 28 = `not used`
Bit 29 = `not used`
Bit 30 = `not used`
Bit 31 = `not used`

Error-Handling

An incorrect label parameter is reported by an error message at the position of the data.

D.10 DAT Label and Data Range

Purpose

DAT labels collect single DAT data items from the ABB automation system for report data files.

Synopsis

#DAT,<*cell*>,<*dir*>,<*netw*>,<*node*>,<*name*>

Description

- **<*cell*>** is the address of the data cell, for example, D10 or REL for Relative.
- **<*dir*>** is the direction of the data range, that is, C for *by column* or R for *by row*.
- **<*netw*>** is the network number of the source DAT object.
- **<*node*>** is the node number of the source DAT object.
- **<*name*>** is the name of the source DAT object.

Example

If you want to obtain data from a DAT named TANK1.MEAN in node 15, network 23, by column, directly below the label cell, define the label as follows:

#DAT,REL,C,23,15,TANK1.MEAN

Return Value

The data range returned from a DAT label is as follows,

Cell address	Data cells	
A1	Valid	Data valid/not valid
A2	Valtype	Value type
A3	Value	Value of the DAT

Figure D-9. Data Range Returned from a DAT Label

where:

- **Valid** is an integer cell that can hold either of the following.

  0 = Not valid

  1 = Valid.

- **Valtype** is an integer cell that can hold one of the following.

  1 = Boolean data (32 bits)

  2 = Integer data (16 bits)

  3 = Long integer data (32 bits)

  4 = Real-value data (32 bits)
- **Value** is always a floating-point number cell.

Error-Handling

An incorrect label parameter is reported by an error message at the position of the data.

D.11 MDAT Label and Data Range

Purpose

MDAT labels collect Multi-DAT data (up to 50 consecutive DAT instances) from the ABB automation system for report data files.

Synopsis

#MDAT,*<cell>,<dir>,<netw>,<node>,<name>,<g1>[,<g2>,...,<g8>]*

Description

- **<*cell*>** is the address of the data cell, for example, HA11 or REL for Relative.
- **<*dir*>** is the direction of the data range, that is, C for *by column* or R for *by row*.
- **<*netw*>** is the network number of the source DAT object.
- **<*node*>** is the node number of the source DAT object.
- **<*name*>** is the name of the first source DAT object.
- **<*gn*>** is the type and number of data items in **group n** specified as <type><numb> where <*type*> is the type of data in group B, I, L or R, and <*numb*> the number of data items in groups 1-50.

NOTE

The sum of <*numb*> in all <*gn*> must not exceed 50, which is the maximum number of DATs in a single multi-DAT request. Store the DATs consecutively without gaps.

Example

If you want to fetch data from 14 consecutive DATs, starting with the one named TANK2.MIN in node 12, network 31, grouped as one Boolean, four reals, one Boolean, four reals and finally four long integers, by row from cell AA12, define the label as follows:

#MDAT,AA12,R,31,12,TANK2.MIN,B1,R4,B1,R4,L4

Return Value

The data range returned from a MDAT label is as follows,

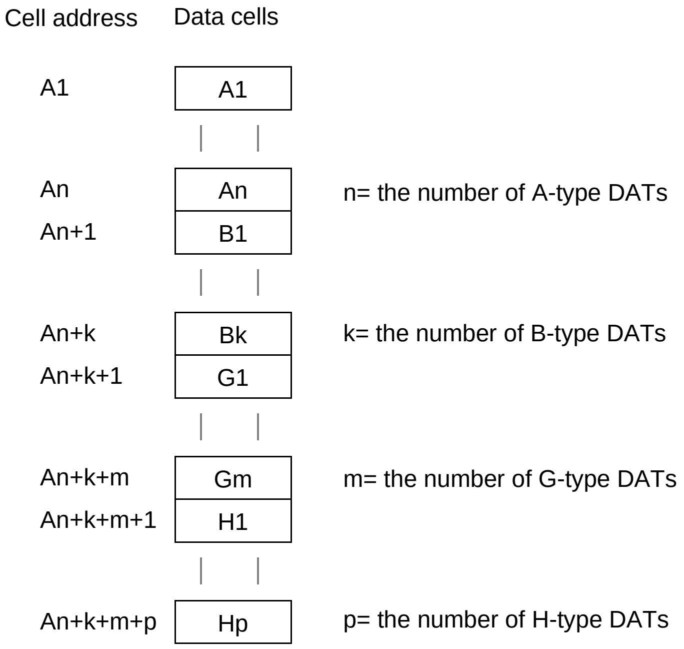

Cell address
Data cells
A1
A1
An
An
n= the number of A-type DATs
An+1
B1
An+k
Bk
k= the number of B-type DATs
An+k+1
G1
An+k+m
Gm
m= the number of G-type DATs
An+k+m+1
H1
An+k+m+p
Hp
p= the number of H-type DATs

Figure D-10. Data Range Returned from a MDAT Label

where the data can be grouped into as many as eight groups in such a way that **n**, **k**, **m** and **p** can each take a value in the range 0-50, but the sum of **n+k+m+p** must not exceed 50. The data cells are all floating-point number cells.

Error-Handling

An incorrect label parameter is reported by an error message at the position of the data.

D.12 TTD Label and Data Range

Purpose

TTD labels collect historical data from the ABB automation system for report data files.

Synopsis

#TTD,<cell>,<dir>,<netw>,<node>,<name>,<var>,<nval>, <ye>,<mo>,<da>,<ho>, <mi>,<se>,[<pres>]

Description

- **<*cell*>** is the address of the data cell, for example, E5 or REL for Relative.
- **<*dir*>** is the direction of the data range, that is, C for *by column* or R for *by row*.
- **<*netw*>** is the network number of the source TTD log.
- **<*node*>** is the node number of the source TTD log.
- **<*name*>** is the name of the source TTD log.
- **<*var*>** is the variable number in the source TTD log (1-127).
- **<*nval*>** is the number of values to fetch, 1-1,000 (see second NOTE below).

Date/time of the oldest value to fetch is specified by:

- **<*ye*>** is the year, in the format aaa, +nn or -nn where aaa = 80-199, denoting 1980-2099 and nn = 0-99.
- **<*mo*>** is the month, in the format aa, +nn or -nn where aa = 1-12 and nn = 0-12.
- **<*da*>** is the day, in the format aa, +nn or -nn where aa = 1-31 and nn = 0-31.
- **<*ho*>** is the hour, in the format aa, +nn or -nn where aa = 0-23 and nn = 0-23.
- **<*mi*>** is the minute, in the format aa, +nn or -nn where aa = 0-59 and nn = 0-59.
- **<*se*>** is the second, in the format aa, +nn or -nn where aa = 0-59 and nn = 0-59.

NOTE

The date/time parameters make it possible to specify both absolute and/or relative times, relative to the system time:

- A value specified without **+/-** denotes an absolute value for date/time in the applicable unit of measure.
- A value specified with **+** defines a positive, relative offset to the current value of date/time in the applicable unit of measure.
- A value specified with **-** defines a negative, relative offset to the current value of date/time in the applicable unit of measure.

- **<*pres*>**is the presentation of time. A stands for ASCII and I for the internal time representation of the spreadsheet software (I is the default).

NOTE

> The number of values to fetch depends on the system. From SuperView 900, a maximum of 1,000 values can be fetched, but a maximum of 402 values can be fetched from process stations (PS).

The date/time of the oldest value to fetch is rounded off upwards by the TTD function, to the next absolute time that matches the time tag of a logged value.

Examples

If you want to obtain the latest 402 historical values from variable 3 in log TANK1.LOG in node 10, network 13, which is a one-hour log with a 15-second logging interval containing 402 values, by row, starting with cell B12, define the label as follows:

#TTD,B12,R,13,10,TANK1.LOG,3,402,+0,+0,+0,-1,+0,+0

This label raises a request for 402 historical values, starting with the one-hour old value (relative to the current time).

TANK3.LOG in node 21, network 13 is a 24-hour log, with a six-minute interval containing 402 values. In this example, it is desirable to collect the historical values of the first eight hours of the current day for the variable at index 7. If you want to orient data by column, starting with cell HG4, define the label as follows:

#TTD,HG4,C,13,21,TANK3.LOG,7,80,+0,+0,+0,0,0,0

This label raises a request for 80 historical values, starting at midnight the current day (a total of eight hours).

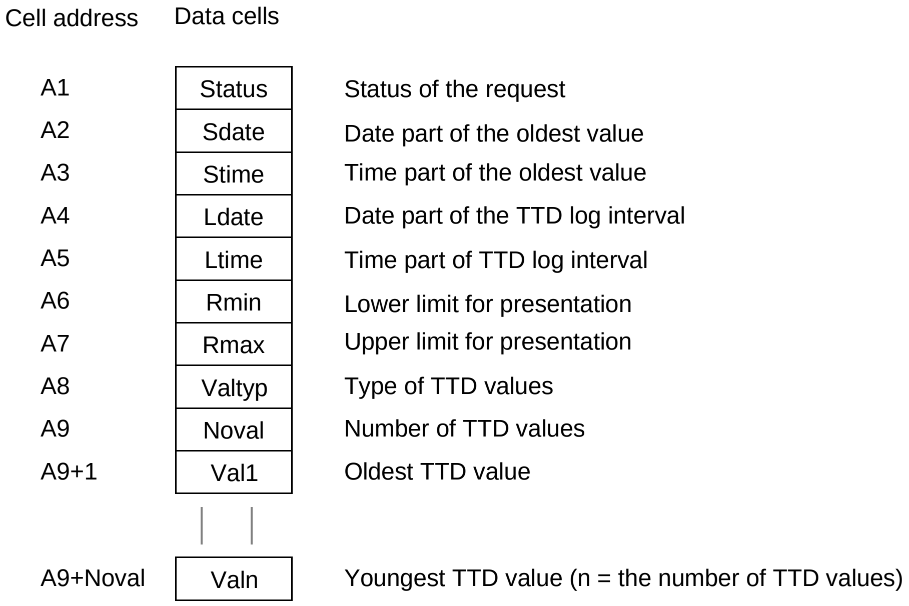
From the same log and the same variable, to collect all logged data, start from the beginning of the previous hour. Do this with a combination of absolute and relative time parameters. Assuming that data is to be oriented by column, directly below the report label, define the label as follows:

#TTD,REL,C,13,21,TANK3.LOG,7,20,+0,+0,+0,-1,0,0

This label raises a request for 20 time-tagged values, starting with the previous hour. For example, if requested at 15:17:10 the data received will cover the period 14:00:00-15:12:00.

Return Value

The data range returned from a TTD label is as follows,



Cell address	Data cells	
A1	Status	Status of the request
A2	Sdate	Date part of the oldest value
A3	Stime	Time part of the oldest value
A4	Ldate	Date part of the TTD log interval
A5	Ltime	Time part of TTD log interval
A6	Rmin	Lower limit for presentation
A7	Rmax	Upper limit for presentation
A8	Valtyp	Type of TTD values
A9	Noval	Number of TTD values
A9+1	Val1	Oldest TTD value
A9+Noval	Valn	Youngest TTD value (n = the number of TTD values)

Figure D-11. Data Range Returned from a TTD Label

where,

- **Status** is an integer cell that can hold one of the following:

  0 = Status ok

  Time range information status

  8 = Time out of range, completely

  9 = Time out of range, oldest limit

  10 = Time out of range, youngest limit

  11 = Time out of range, both limits

  Error status

  2 = Log not found

  3 = Log not implemented

  4 = Variable not found

  5 = Variable not implemented

- **Sdate** is a floating-point number cell or a label cell.
- **Stime** is a floating-point number cell or a label cell.
- **Ldate** is a floating-point number cell or a label cell.

- **Ltime** is a floating-point number cell or a label cell.
- **Rmin** is a floating-point number cell.
- **Rmax** is a floating-point number cell.
- **Valtyp** is an integer cell.
- **Noval** is an integer cell.

TTD values are always floating-point number cells. If a TTD value is out of range, it will be marked as ERR.

Cells containing times hold data in either ASCII or internal representation for ease of use. Cells containing dates hold data in the internal representation, also for ease of use. They are formatted by default to be legible.

NOTE

If Status is in error, the contents of the rest of the data range may be unpredictable.

Error-Handling

An incorrect label parameter is reported by an error message at the position of the data.

D.13 TTDO Label and Data Range

Purpose

TTDO labels collect historical data from the ABB automation system for report data files. A TTDO data request accesses the desired log via the name of the object in the log rather than via the name of the log directly, as is the case with TTD labels.

Synopsis

#TTDO,<cell>,<dir>,<netw>,<node>,<name>,<prop>,<int>,<func>,<nval>,
<ye>,<mo>,<da>,<ho>,<mi>,<se>,[<pres>]

Description

- ***<cell>*** is the address of the data cell, for example, D2 or REL for Relative.
- ***<dir>*** is the direction of the data range, that is, C for by column or R for by row.
- ***<netw>*** is the network number of the source TTD log.
- ***<node>*** is the node number of the source TTD log.
- ***<name>*** is the name of the logged object.
- ***<prop>*** is the property of the object logged (see Table D-2).
- ***<int>*** is the log interval expressed in value and unit (see Table D-3).
- ***<func>*** is the log function 1-4 (see Table D-4).
- ***<nval>*** is the number of values to fetch, 1-1,000 (see NOTE).

The date/time of the oldest value to fetch is specified by:

- ***<ye>*** is the year, in the format aaa, +nn or -nn where aaa = 80-199, denoting 1980-2099 and nn = 0-99.
- ***<mo>*** is the month, in the format aa, +nn or -nn where aa = 1-12 and nn = 0-12.
- ***<da>*** is the day, in the format aa, +nn or -nn where aa = 1-31 and nn = 0-31.
- ***<ho>*** is the hour, in the format aa, +nn or -nn where aa = 0-23 and nn = 0-23.
- ***<mi>*** is the minute, in the format aa, +nn or -nn where aa = 0-59 and nn = 0-59.
- ***<se>*** is the second, in the format aa, +nn or -nn where aa = 0-59 and nn = 0-59.

NOTE

The format of the date/time variables makes it possible to define both absolute and/or relative times relative to the system time.

- A value specified without **+/-** defines an absolute time in the applicable unit of measure
- A value specified with **+** defines a relative increment to the current value in the applicable unit of measure

  - A value specified with **-** defines a relative decrement from the current value in the applicable unit of measure

- <***pres***>is the presentation of time. A stands for ASCII and I stands for the internal representation of the spreadsheet software (I is the default).

Table D-2. Properties for Different Object Types

Object type	Property logged	Short name
AI	Value	MV
AO	Value	OUT
DI	Value	MV
DO	Value	OUT
PIDCON	Measured value Setpoint Output signal Deviation Actual or position	MV SP OUT DEV ACT
MANSTN	Measured value Output signal Actual or position	MV OUT ACT
RATIOSTN	Measured value Ratio Output signal	MV RAT OUT
GENCON	Measured value Setpoint Output signal	MV SP OUT
GENBIN	Measured value	MV
GENUSD	Measured value Mea. value intl Setpoint	MV MV2 SP
GENNOV	Measured value Setpoint Output signal Ext. setpoint Feed forward value	MV SP OUT ESP FF
MOTCON	Motor current	CURR

Table D-2. Properties for Different Object Types

VALVECON	Valve position	VALVP
DAT	Real value Integer long value Integer value Boolean value*	R IL I Bn(1)

(1)The Boolean value is Bn, where n=1-32

Table D-3. Log Intervals

Log interval			
(sec)	**(min)**	**(hour)**	**(week)**
	1M	1H	1W
	2M	2H	
	3M	3H	
	4M	4H	
	5M	6H	
	6M	8H	
10S	10M	12H	
12S	12M	24H	
15S	15M		
20S	20M		
30S	30M		

Table D-4. Log Functions

Log function	**Short name**
Mom value	MOM
Mean value	MEAN
Max value	MAX
Min value	MIN

NOTE

The number of values that can be fetched depends on the system. You can fetch a maximum of 1,000 values from SuperView 900, but a maximum of only 402 values from process stations (PS).

The date/time of the oldest value to fetch is rounded up by the TTD function in the ABB automation system to the next absolute time that matches the time tag of a logged value.

Example

If you want to obtain the latest 120 mean values for PUMP1.SP from a one-minute log in node 12, network 20, by column, starting with cell D12, define the label as follows:

#TTDO,D12,C,20,12,PUMP1.SP,OUT,1M,MEAN,120,+0,+0,+0,-2,+0,+0

This label raises a request for 120 historical values, starting with the value that is two hours old (relative to the current time).

Return Value

The data range is identical to that of the TTD label. Refer to Section 6.1.4.11, TTD Label and Data Range.

D.14 COM Label and Data Range

Purpose

COM labels collect communication status information from the nodes of the ABB automation system for report data files.

Synopsis

#COM,<*cell*>,<*dir*>,<*netw*>,<*node*>

Description

- **<*cell*>** is the address of the data cell, for example, A5 or REL for Relative.
- **<*dir*>** is the direction of the data range, that is, C for *by column* or R for *by row*.
- **<*netw*>** is the network number of the source station.
- **<*node*>** is the node number of the source station.

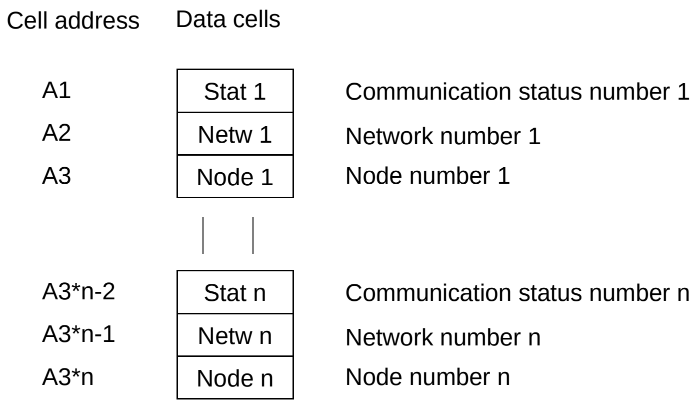
Example

If you want to obtain communication status information from node 21 in network 13, by column, directly below the label, define the label as follows:

#COM,REL,C,13,21

Return Value

The data range returned from a COM label is as follows,



Cell address	Data cells	
A1	Stat 1	Communication status number 1
A2	Netw 1	Network number 1
A3	Node 1	Node number 1
A3*n-2	Stat n	Communication status number n
A3*n-1	Netw n	Network number n
A3*n	Node n	Node number n

Figure D-12. Data Range Returned from a COM Label

where:

- **Netw n** are the network numbers in integer cells.
- **Node n** are the node numbers in integer cells.
- **Stat n** are floating-point number cells defined as bitmaps, each bit holding different status information:

  Bit 0 = NO COMMUNCATION (0 = Communication OK)
  Bit 1 = `not used`
  Bit 2 = MB200 BUS A DOWN (for MB200 bus master only)
  Bit 3 = MB200 BUS B DOWN (for MB200 bus master only)
  Bit 4 = MB200 NODE BUS MASTER (for MB200 only)
  Bit 5 = MB100 LINK
  Bit 6 = MB200 LINK
  Bit 7 = MB300 LINK
  Bit 8 = `not used`
  Bit 9 = CNC LINK
  Bit 10 = MVI LINK
  Bit 11 = GCOM LINK
  Bit 12 = RCOM LINK
  Bit 13 = `not used`
  Bit 14 = `not used`
  Bit 15 = `not used`

NOTE

The following rules apply to the possible combinations of status bits in the Stat bitmap:

- **Communication status bit b0**. Status bit b0 is independent of all other bits and can be set independently of all other bits.
- **Communication link type bits b5 to b12.** Only one of the status bits b5 to b12 can be set.
- **MB 200 status bits b2, b3 and b4.** The MB 200 status bits b2, b3 and b4 are only relevant if bit b6 (MB 200 link) is set. The MB 200 status bits b2 and b3 are only relevant if bit b4 (MB 200 bus master) is set.

Error-Handling

An incorrect label parameter will be reported by an error message at the position of the data.

D.15 EVL Label and Data Range

Purpose

EVL labels collect process events from a MV 800/1 for report data files. Requests may include the time of the first event to be reported or specify covering events not previously reported.

Synopsis

#EVL,*<cell>,<netw>,<node>,<subsy..>,<nevent>,[<pres>,[<ye>,<mo>, <da>,<ho>, <mi>,<se>]*]

Description

- <***cell***> is the address of the data cell, for example, A5 or REL for Relative. The data is always arranged by column
- <***netw***> is the network number of the source operator station (OS).
- <***node***> is the node number of the source operator station (OS).
- <***subsy..***> is the subsystems of the events to fetch. 0 stands for all subsystems, 1-16 specifies a unique subsystem which could be combined as a+b+c, etc.
- <***nevent***> is the number of requested events, 1-600 (see NOTE).
- <***pres***> is the presentation of time. A stands for ASCII and I stands for the internal representation of the spreadsheet software (I is the default).

The EVL label has two functions:

1. If date/time parameters are omitted, only those events that have not been reported previously are reported.
2. If date/time parameters are given, all existing events falling within the specified time frame are reported, including those reported previously. This function does not affect the events covered by Function 1, above.

NOTE

The maximum number of events per request for Function 2, above, is 30. The number of events received may be less than the number requested if the number of events occurring is less than the number requested.

The date/time of the oldest event to fetch is specified as follows:

- **<*ye*>** is the year, in the format aaa, +nn or -nn where aaa = 80-199, denoting 1980-2099, and nn = 0-99.
- **<*mo*>** is the month, in the format aa, +nn or -nn where aa = 1-12 and nn = 0-12.
- **<*da*>** is the day, in the format aa, +nn or -nn where aa = 1-31 and nn = 0-31.
- **<*ho*>** is the hour, in the format aa, +nn or -nn where aa = 0-23 and nn = 0-23.
- **<*mi*>** is the minute, in the format aa, +nn or -nn where aa = 0-59 and nn = 0-59.
- **<*se*>** is the second, in the format aa, +nn or -nn where aa = 0-59 and nn = 0-59.

NOTE

The date/time parameters make it possible to specify both absolute and/or relative times, relative to the current system time:

- A value specified without **+/-** defines an absolute value in the applicable unit of measure.
- A value specified with **+** defines a positive, relative offset to the current value in the applicable unit of measure.
- A value specified with **-** defines a negative, relative offset to the current value in the applicable unit of measure.

Example

If you want to obtain a maximum of 400 event messages according to Function 1 from the subsystems 1, 2 and 3 in node 1, network 13, by column, directly below the label with time in ASCII form, define the label as follows:

#EVL,REL,13,1,1+2+3,400,A

Return Value

The data range returned from an EVL label is as follows:

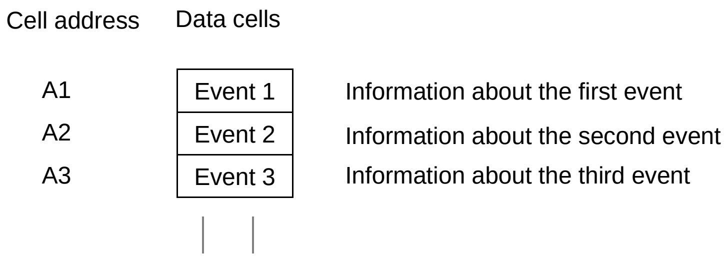

Cell address
Data cells
A1
A2
A3
Event 1
Event 2
Event 3
Information about the first event
Information about the second event
Information about the third event

Figure D-13. Data Range Returned from an EVL Label

Each event message contains the following,

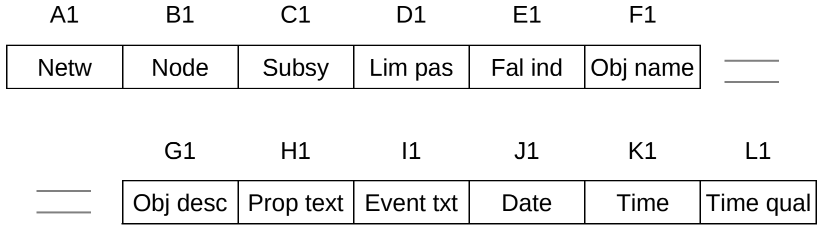


Figure D-14. The Content of each Event Message

where:

- **Netw** is an integer cell (3).
- **Node** is an integer cell (3).
- **Subsy** (subsystem) is an integer cell (3).
- **Lim pas** (limit passed) is a label cell (3).
- **Fal ind** (fault indication) is a label cell (2).
- **Obj name** (object name) is a label cell (9).
- **Obj desc** (object description) is a label cell (3).
- **Prop txt** (property text) is a label cell (9).
- **Event txt** (event text) is a label cell (15).
- **Date** is an integer cell (7).
- **Time** is an integer or label cell (13).
- **Time qual** (time quality) is a label cell (2).

The numbers inside parentheses are the recommended cell widths to ensure the layout is legible on an 80-column screen or printer. You can define these cell widths by formatting the column width in the worksheet.

NOTE

Obj name and **Obj desc** can be up to 20 characters long.

Error-Handling

An incorrect label parameter is reported by an error message at the position of the data.

D.16 EVLC Label and Data Range

Purpose

EVLC labels collect process event messages cyclically from the MV 800/1 for report data files. EVLC labels can create up to 999 different, user-defined report data files, which are given the extensions .001, .002, .003-.999.

Synopsis

#EVLC,<*cell*>,<*netw*>,<*node*>,<*subsy..*>,<*nevent*>,<*func*>,<*nfiles*>, [<*pres*>]

Description

- **<*cell*>** is the address of the data cell, for example, A5 or REL for Relative. The data is always arranged by column.
- **<*netw*>** is the network number of the source station
- **<*node*>** is the node number of the source station
- **<*subsy..*>** are the subsystems of the events to fetch. 0 stands for all subsystems. 1-16 specifies a unique subsystem which could be combined as a+b+c, etc.
- **<*nevent*>** is the number of events to fetch, 1-600.
- **<*func*>** is the function of the EVLC, C or S. C indicates report data files are stored in a circular buffer with <nfiles> instances. S indicates that the maximum number of report data files is <nfiles>. On arrival at this number, execution stops.
- **<*nfiles* >** is the number of data files, 1-999, which brings the maximum number of stored events to 599,400.
- **<*pres*>** is the presentation of time. A stands for ASCII and I stands for the internal representation of the spreadsheet software (I is the default).

NOTE

EVLC labels cannot be combined with other labels. You can use only one EVLC label per report definition file, so you can fetch process events from only one node at a time.

Example

If you want to obtain event messages cyclically from all the subsystems of node 1 in network 13, by column, starting at A10 (cells A10, A11, A12, etc.) in a circular buffer consisting of 10 files with 600 events in each and with the internal presentation of time, define the label as follows:

#EVLC,A10,13,1,0,600,C,10

Return Value

The data range is identical to that of EVL. See Section D.15 EVL Label and Data Range, EVL Label and Data Range.

Error-Handling

The error-handling is supplemented with an error message to the DOS operating system.

D.17 TRIGG Label and Data Range

Purpose

TRIGG labels trigger compilation and output of reports on receipt of a trigger DS or TS from the ABB automation system. The triggered report may contain data for dispatch (DS or TS) as well as requests for data. The report definition file may also contain report labels which are not connected to any TRIGG label. These labels are the first ones to execute, resulting in the creation of an ordinary report data file (report definition file.WKD (or .XLD)).

For each activated trigger, a report data file is created, which is named after the TRIGG label. The extensions to the file name are .001, .002, etc., depending on the number of times the trigger is released during execution of the report. The report executes until all the TRIGG labels are timed out or until you terminate the program. In this way, the trigger function lets you create event-driven reports which are controlled by the application in the ABB automation system.

Synopsis

#TRIGG,*<cell>,<dir>,<netw>,<node>,<type>,<id>,<outfile>,[[[<ye>, <mo>,<da>, <ho>,<mi>,<ss>], <num>], <nb>, <ni>, <nl>, <nr>]*

Description

- **<*cell*>** is the address of the trigger (DS or TS) data cell, for example, B8 or REL for Relative.
- **<*dir*>** is the direction of the data range, that is, C for *by column* or R for *by row*
- **<*netw*>** is the network number of the trigger.
- **<*node*>** is the node number of the trigger.
- **<*type*>** is the type of trigger (DS, TS, DT or TT) where:

  DS is a DSC trigger.

  TS is a TSC trigger.

  DT is a DSC terminate trigger.*

  TT is a TSC terminate trigger.*

  *Terminate triggers will, on arrival, terminate the report execution. Refer to Section 3.1.2 Report Scheduling with SuperView 100 Batch Files, for more information on how you can use terminate triggers for batch control.

  NOTE

  Report labels connected to terminate triggers are never executed.

- **<*id*>** is the identity number of the trigger (identity number of the DS or TS).
- **<*outfile*>** is the data file for the report. The file name (including path information) cannot exceed 76 characters. Give the name without an extension.

Use date/time to specify the trigger's time-out time:

- **<*ye*>** is the year, in the format 'aaa', '+nn' or '-nn' where aaa = 80 - 199, denoting 1980-2099, and nn = 0-99.
- **<*mo*>** is the month, in the format aa, +nn or -nn where aa = 1-12 and nn = 0-12.
- **<*da*>** is the day, in the format aa, +nn or -nn where aa = 1-31 and nn = 0-31.
- **<*ho*>** is the hour, in the format aa, +nn or -nn where aa = 0-23 and nn = 0-23.
- **<*mi*>** is the minute, in the format aa, +nn or -nn where aa = 0-59 and nn = 0-59.
- **<*se*>** is the second, in the format aa, +nn or -nn where aa = 0-59 and nn = 0-59.

NOTE

You can specify both absolute and/or relative times, relative to the current system time, using the date/time parameters:

- Specify a value without **+/-** to define an absolute value in the applicable unit of measure.
- Specify a value with **+** to define a positive, relative offset to the current value in the applicable unit of measure.
- Specify a value with **-** to define a negative, relative offset to the current value in the applicable unit of measure.

- **<*num*>** is the number of times the TRIGG is allowed to release, 1-999 or no limit (-1). The default setting is no limit. In case of terminate triggers, the value of <*num*> is ignored since these triggers are released only once during a report execution.
- **<*nb*>**, **<*ni*>**, **<*nl*>** and **<*nr*>** are used only with DS triggers. They specify the number of different data types included in the DS and are set to 1 by default.
- **<*nb*>** is the number of Boolean data items, 0-24.
- **<*ni*>** is the number of integer data items, 0-24.
- **<*nl*>** is the number of long integer data items, 0-24.
- **<*nr*>** is the number of real-value data items, 0-24.

NOTE

If date/time is omitted, or if the date/time has passed, the trigger function continues until the number of times specified in <num> is reached. If <num> is set to No Limit (-1), the function continues indefinitely. The only way you can terminate it is by generating a Break interrupt (DOS version) or by using **Stop report exec** from the **Control** menu (Windows version).

The EVLC label is not allowed in triggered reports.

Example

Suppose you want to obtain a trigger from nodes 21, 22 or 23 in network 13 and to write the report to files named TR1321.00x, TR1322.00x and TR1323.00x, respectively. Suppose further that you have one DS trigger with the identity number 2, which should release only one time within the time-out interval (five hours). The DS trigger consists of four values (one of each data type), so you can use the default values.

Other triggers include a TS with the identity number 58 and a TS with the identity number 14. The two TS triggers should time out after 10 and eight hours, respectively.

There is also a DSC terminate trigger with the identity number 1. It consists of only one integer value and the time-out time is set to five years, which, in practical terms, sets it indefinitely.

These prerequisites give rise to the following trigger labels and report definitions:

```
#AI,C14,R,13,25,TANK6.SP

#TRIGG,A10,C,13,21,DS,2,TR1321,+0,+0,+0,+5,+0,+0,1
#DAT,C12,R,13,20,TANK1.MEAN
#AI,C14,R,13,21,TANK1.SP
#DI,C16,R,13,21,FAN3.SW

#TRIGG,A4,R,13,22,TS,58,TR1322,+0,+0,+0,+10,+0,+0
#DAT,C12,R,13,22,TANK3.MIN
#AI,C14,R,13,20,PUMP2.SP
#DI,C16,R,13,20,VALVE1.POS
#TRIGG,C10,C,13,23,TS,14,TR1323,+0,+0,+0,+8,+0,+0
#AO,C12,C,13,18,PUMP2.OUT
#DO,C14,C,13,20,VALVE3.POS
#DSC,C5,C,13,12,7,S,2,4,4,12

#TRIGG,A10,C,13,21,DT,1,STOP,+5,+0,+0,+0,+0,+0,1,0,1,0,0

#AO,C12,C,13,15,TANK8.OUT
```

NOTE

The last DSC is a sending data set, where the following sending data are needed: cells C5 and C6 must contain the two Booleans, cells C7 to C10 must contain the four integers, cells C11 to C14 must contain the four long integers and cells C15 to C26 must contain the 12 real values.

End the report request definition with an empty line or a new TRIGG label and place it directly below the TRIGG label it corresponds to. The first, AI (TANK6), and the last, AO (TANK8), are not connected to any TRIGG label and are executed in a report before the TRIGG reports.

Return Value

The output depends on the corresponding report data. See the data range descriptions of the other labels.

Error- Handling

An incorrect label parameter is reported by an error message at the position of the data. There is also an error message at the DOS level.

Appendix E SuperView 100 Windows Batch File Commands

E.1 ABSWAIT

Purpose

ABSWAIT will hold the next command execution to an absolute time.

Synopsis

ABSWAIT <hh>:<mm>

Description

- **<hh>** is the absolute time in hours (00 - 23)**.**
- **<mm>** is the absolute time in minutes (00 - 59).

NOTE

If the time is older than the current time, the ABSWAIT function will wait until the next day.

Example

To hold the execution of the next batch command until 12 o'clock, use the following

ABSWAIT 12:00

Return Value

No return value.

Error- Handling

If an error occurred the **ERROR** flag will be set true and an error message will be displayed in the error log window.

E.2 RELWAIT

Purpose

RELWAIT will hold the next command execution a relative time.

Synopsis

RELWAIT <hh>:<mm>

Description

- **<hh>** is the relative time in hours (00 - 23)**.**
- **<mm>** is the relative time in minutes (00 - 59).

Example

To hold the execution of the next batch command 1 hour and 30 minutes, use the following

RELWAIT 01:30

Return Value

No return value.

Error- Handling

If an error occurred the **ERROR** flag will be set true and an error message will be displayed in the error log window.

E.3 EXEC

Purpose

EXEC is used to start other Windows or DOS (.EXE) programs.

Synopsis

EXEC <*application*>

Description

- **<application>** is the path and file name of the Windows or DOS (.EXE) program**.**

NOTE

This program will be started as a new process in MS-Windows, and the next line in the batch file will be executed immediately after program start-up.

Example

To start up Microsoft Excel from a SV 100 batch file do the following:

```
EXEC c:\excel\excel.exe
```

Return Value

No return value.

Error- Handling

If an error occurred the **ERROR** flag will be set true and an error message will be displayed in the error log window.

E.4 SV100

Purpose

SV100 submits a report definition file to the report execution part of SuperView 100.

Synopsis

SV100 <*file*> </*qualifiers*>

Description

- **<file>** is the path and file name of the WK1 or XLS report definition file.
- **</*qualifiers*>** are different qualifiers provided by the SuperView 100 report software and are described in detail in Section 4.2.1.2 The Command Line Interface.

NOTE

The next line in the batch file will not be executed until the report execution is completed.

Example

This will start a report execution of report1.xls do the following:

```
SV100 c:\excel\report1.xls
```

Return Value

If the report execution was terminated by a trigger (see Section D.17 TRIGG Label and Data Range) the **TRIGGIDENT** flag will be set to the identity of the terminated trigger.

Error- Handling

If an error occurred the **ERROR** flag will be set true and an error message will be displayed in the error log window.

E.5 GOTO

Purpose

GOTO is used to make an absolute jump in the batch file. This can be used to create command loops.

Synopsis

GOTO <string>

Description

- **<string>** is the string reference from where the batch file execution will continue.

Example

To make a jump in the batch file to **:start** reference do the following:

```
:start
...other batch commands...
GOTO start
```

Return Value

No return value.

Error- Handling

If an error occurred the **ERROR** flag will be set true and an error message will be displayed in the error log window.

E.6 :*<string>*

Purpose

:<string> is a reference used by **GOTO** and **GOSUB** to make an absolute jump in the batch file. This can be used to create subroutines or command loops.

Synopsis

:<string>

Description

- **<string>** is the string reference from where the batch file execution will continue.

Example

See **GOTO** or **GOSUB** example.

Return Value

No return value.

Error- Handling

No error handling.

E.7 GOSUB

Purpose

GOSUB is used to make an absolute jump in the batch file. This can be used to create subroutines in the batch file, which for example can be called when a specific terminating trigger has released. When the subroutine is completed it must end with a **RETURN**, and the batch file execution will then continue with the next batch command after **GOSUB**.

Synopsis

```
GOSUB <string>
```

Description

- **<string>** is the string reference from where the batch file execution will continue.

NOTE

Define subroutines in batch files after the batch main loop to prevent these routines from being executed during normal runtime.

Example

To make a jump in the batch file to **:start** reference do the following:

```
:start
GOSUB sub1
...other batch commands...
GOTO start
:sub1
...other batch commands...
RETURN
```

Return Value

No return value.

Error- Handling

If an error occurred the **ERROR** flag will be set true and an error message will be displayed in the error log window.

E.8 RETURN

Purpose

RETURN defines the end of a subroutine in the batch file. After this line, the batch execution continues with the next line after the **GOSUB** call.

Synopsis

RETURN

Description

See Purpose.

Example

See **GOSUB**.

Return Value

No return value.

Error- Handling

No error handling.

E.9 IF TRIGGIDENT=

Purpose

IF TRIGGIDENT= is used to check the identity of terminated TRIGG labels. Terminate triggers (refer to the TRIGG label description in Section D.17 TRIGG Label and Data Range) set the value of **TRIGGIDENT** to the identity of the trigger terminated so you can control batch file execution based on ABB automation system events. If the right trigger has terminated the next batch command will be executed, otherwise this command will be omitted. The batch execution will then continue with the next batch command.

Synopsis

IF TRIGGIDENT=<integer value>

Description

- **<integer value>** is the identity of the terminated trigger.

Example

To check if a specific terminate trigger has released do the following:

```
:start
SV100 TRIGGERS.XLS
IF TRIGGIDENT=10
GOSUB sub2
GOTO start
:sub2
...other batch commands...
RETURN
```

If the terminated trigger has identity 10 then the subroutine sub2 is called. Otherwise, or when the subroutine is returned, the batch execution jumps to **:start**

Return Value

Return true if **TRIGGIDENT** equals **<integer value>**.

Error- Handling

If an error occurred the **ERROR** flag will be set true and an error message will be displayed in the error log window.

E.10 IF ERROR

Purpose

IF ERROR is used to check if the previous batch command failed. The user can thereby, for example, exit from the batch file execution if a batch command fails. If the **ERROR** flag is set true then the next batch command will be executed, otherwise this command will be omitted. The batch execution will then continue with the next batch command.

Synopsis

```
IF ERROR
```

Description

See Purpose.

Example

To stop the batch execution if the report execution of report.xls fails do the following:

```
SV100 report.xls
IF ERROR
EXIT
```

Return Value

Return true if an error has occurred in the previous batch command.

Error- Handling

No error handling.

E.11 EXIT

Purpose

EXIT will immediately exit you from the batch file execution. Use the **Control** menu and the **Start batch execution** alternative to continue the batch execution, or reset the batch file to the beginning with the **Reset batch execution** alternative.

Synopsis

EXIT

Description

See Purpose.

Example

See **IF ERROR**.

Return Value

No return value.

Error- Handling

No error handling.

E.12 ;<comment>

Purpose

;<comment> is used for comments in the batch file.

Synopsis

```
;<comment>
```

Description

<comment> is the comment string.

Example

To make a comment in the batch file do the following:

```
; This is a comment
```

Return Value

No return value.

Error- Handling

No error handling.

Appendix F Syntax for DDE Subscriptions

F.1 General

General DDE Syntax

All applications using DDE to exchange data must include the following elements.

- **Application name** (the name of the application)
- **Topic name** (the basic data grouping)
- **Item description** (the data within the topic)

The application name for the SuperView 100 Report software is **SV100.**

NOTE

The SuperView 100 Report software must be running before you can start to use DDE. This to ensure a proper GCOM connection before data subscription starts. **The DDE client can not start up the SuperView 100 Report software.**

The topic name specifies the object type for the data to be collected, that is, one of AI, AO, DI, DO, DAT, MDAT, DSC or TSC.

The topic name SYSTEM is used when sending commands to the SuperView 100 Report software.

The table below shows the topic names supported using DDE:

Table F-1. Supported Topic Names Using DDE

Topic	Description
DSC	Data Set (both receipt and dispatch)
TSC	Text Set (both receipt and dispatch)
AI	AI data
AO	AO data
DI	DI data
DO	DO data
DAT	DAT data
MDAT	Set of consecutive DAT instances
SYSTEM	Used when sending commands to SV100

The item description specifies which object and what data should be collected from the object. Each type of object has its own set of parameters and entry syntax. This syntax and the layout of the corresponding data range are described in detail in the following sections.

The data ranges hold the data requested in the item description and collected for the report objects. Data layout is fixed for each type of item.

The description of the DDE link definition is supplemented by examples. The corresponding data range layouts are described with cell A1 as anchor cell and with the data organized by column (all data in the same column).

The item description used when sending commands to the SuperView 100 Report software is identical to the parameters and qualifiers provided by the command line interface. For more information, please refer to Section 4.2.1.2 The Command Line Interface.

F.2 AI Objects

Purpose

AI objects collect analog input data from the ABB automation system.

Synopsis

The topic name used to collect analog input data is AI. Use the following syntax for the item description for AI objects:

objectname[\objectname...]\dataitem

Description

- ***objectname*** is the name of the object(s) from which data is collected. You can request a number of objects in each DDE request.
- ***dataitem*** specifies the data to be collected, common to all objects in the item description, and can be any of the following:

  Full defines all the data items that can be collected for the object.

  All defines all the data items that can be collected for the object, except Unit.

  Value is the object data value.

  Status is a bitmap with some status information about the object.

  Rmin is the value for minimum range of the AI object.

  Rmax is the value for maximum range of the AI object.

  Unit is a string holding the unit of the value for AI objects (for example, %).

The item description is limited to 1000 characters.

Example

If you want to obtain the values for three AI objects (AIC21_1, AIC21_2 and AIC21_3) using Microsoft Excel, use the following DDE link:

SV100|AI!AIC21_1\AIC21_2\AIC21_3\VALUE

Return Value

The maximum data range returned from an AI object is as follows,

Cell address	Data cells	
A1	Status	Status of the request
A2	Value	Value of the AI

Figure F-1. Maximum Data Range Returned from an AI Object

where:

- **Status** is a long integer number cell defined as a bitmap where each bit holds different status information:

  Bit 0 = IMPLEMENTED
  Bit 1 = ERROR
  Bit 2 = UPDATED
  Bit 3 = UPD_BLK
  Bit 4 = MAN_ENTRY
  Bit 5 = SELECTED
  Bit 6 = NORMAL_OBJ_TREAT
  Bit 7 = H2_REP_FAIL_CTRL
  Bit 8 = ABOVE_HI_LIM2
  Bit 9 = ABOVE_HI_LIM1
  Bit 10 = BELOW_LO_H1
  Bit 11 = BELOW_LO_H2
  Bit 12 = ALARM_UNACK
  Bit 13 = REPEAT_FAIL_BLK
  Bit 14 = DISTURBANCE
  Bit 15 = OVERFLOW
  Bit 16 = ALARM_BLK
  Bit 17 = ALARM_PERIOD_BLK
  Bit 18 = PRINT_BLK
  Bit 19 = REPEAT_FAIL_CTRL
  Bit 20 = SYNC
  Bit 21 = SYNC_UPD
  Bit 22 = NOERR_AT OVF
  Bit 23 = TESTED
  Bit 24 = ACC_ERR
  Bit 25 = ACC_LOCK
  Bit 26 = L2_REP_FAIL_CTRL

Bit 27 = ER_REP_FAIL_CTRL
Bit 28 = REPEAT_FAIL_CTRL
Bit 29 = AI_ACTION
Bit 30 = AI_RETRY
Bit 31 = ERR_CTRL

- **Value** is a floating-point number cell.
- **Rmax** is a floating-point number cell.
- **Rmin** is a floating-point number cell.
- **Unit** is a label cell.

F.3 AO Objects

Purpose

AO objects collect analog output data from the ABB automation system.

Synopsis

The topic name used for collection of analog input data is AO. Use the following syntax for the item description for AO objects:

objectname[\objectname...]\dataitem

Description

- ***objectname*** is the name of the object(s) from which data is collected. You can request a number of objects in each DDE request.
- ***dataitem*** specifies the data to be collected, common to all objects in the item description, and can be any of the following:

  Full defines all the data items that can be collected for the object

  All defines all the data items that can be collected for the object, except Unit.

  Value is the object data value.

  Status is a bitmap with some status information about the object.

  Rmin is the value for minimum range of the AO object.

  Rmax is the value for maximum range of the AO object.

  Unit is a string holding the unit of the value for AO objects (for example, %).

The item description is limited to 1000 characters.

Example

If you want to obtain the values for three AO objects (AOC21_1, AOC21_2 and AOC21_3) using Microsoft Excel, use the following DDE link:

SV100|AO!AOC21_1\AOC21_2\AOC21_3\VALUE

Return Value

The maximum data range returned from an AO object is as follows,

Cell address	Data cells	
A1	Status	Status of the request
A2	Value	Value of the AO
A3	Rmin	Min. range of the AO value
A4	Rmax	Max. range of the AO value
A5	Unit	Unit of the AO

Figure F-2. Maximum Data Range Returned from an AO Object

where:

- **Status** is a long integer number cell defined as a bitmap, with each bit holding different status information:

  Bit 0 = IMPLEMENTED
  Bit 1 = ERROR
  Bit 2 = UPDATED
  Bit 3 = OUTP_BLK
  Bit 4 = not used
  Bit 5 = SELECTED
  Bit 6 = NORMAL_OBJ_TREAT
  Bit 7 = MAN_MODE
  Bit 8 = ON_MAX_LIM
  Bit 9 = not used
  Bit 10 = not used
  Bit 11 = ON_MIN_LIM
  Bit 12 = ALARM_UNACK
  Bit 13 = not used
  Bit 14 = not used
  Bit 15 = not used
  Bit 16 = ALARM_BLK
  Bit 17 = ALARM_PERIOD_BLK
  Bit 18 = PRINT_BLK
  Bit 19 = OUTP_RESTART
  Bit 20 = LOCK
  Bit 21 = USE_MAX_LIM
  Bit 22 = USE_MIN_LIM

Bit 23 = TESTED
Bit 24 = `not used`
Bit 25 = `not used`
Bit 26 = LINKED
Bit 27 = `not used`
Bit 28 = `not used`
Bit 29 = `not used`
Bit 30 = `not used`
Bit 31 = `not used`

- **Value** is a floating-point number cell.
- **Rmax** is a floating-point number cell.
- **Rmin** is a floating-point number cell.
- **Unit** is a label cell.

F.4 DI Objects

Purpose

DI objects collect digital input data from the ABB automation system.

Synopsis

The topic name used for collection of analog input data is DI. Use the following syntax for the item description for DI objects:

objectname[\objectname...]\dataitem

Description

- ***objectname*** is the name of the object(s) from which data is collected. You can request a number of objects in each DDE request.
- ***dataitem*** *s*pecifies the data to be collected, common to all objects in the item description, and can be any of the following:

  All, **Full** define all the data items that can be collected for the object.

  Value is the object data value.

  Status is a bitmap with some status information about the object.

The item description is limited to 1000 characters.

Example

To obtain the values for three DI objects (DIC21_1, DIC21_2 and DIC21_3) using Microsoft Excel, use the following DDE link:

SV100|DI!DIC21_1\DIC21_2\DIC21_3\VALUE

Return Value

The maximum data range returned from a DI object is as follows,

Cell address	Data cells	
A1	Status	Status of the request
A2	Value	Value of the DI

Figure F-3. Maximum Data Range Returned from a DI Object

where:

- **Value** is an long integer cell.
- **Status** is a long integer number cell defined as a bitmap where each bit holds different status information.

Bit 0 = IMPLEMENTED
Bit 1 = ERROR
Bit 2 = UPDATED
Bit 3 = UPD_BLK
Bit 4 = MAN_ENTRY
Bit 5 = SELECTED
Bit 6 = NORMAL_OBJ_TREAT
Bit 7 = ERR_CTRL
Bit 8 = VALUE
Bit 9 = SEC_VALUE
Bit 10 = NORM_POSN
Bit 11 = SEL_NORM_POSN
Bit 12 = ALARM_UNACK
Bit 13 = REPEAT_FAIL_BLK
Bit 14 = DISTURBANCE
Bit 15 = CALC_VALUE
Bit 16 = ALARM_BLK
Bit 17 = ALARM_PERIOD_BLK
Bit 18 = PRINT_BLK
Bit 19 = REPEAT_FAIL_CTRL
Bit 20 = LOCK
Bit 21 = INVERTED
Bit 22 = DUAL_IND
Bit 23 = TESTED
Bit 24 = `not used`

Bit 25 = V9_CONTROL
Bit 26 = LINKED
Bit 27 = DI_RETRY
Bit 28 = DIC_IND_RED
Bit 29 = DI_ACTION
Bit 30 = DIC_IND_YELLOW
Bit 31 = DIC_IND_GREEN

F.5 DO Objects

Purpose

DO objects collect digital input data from the ABB automation system.

Synopsis

The topic name used for collection of analog input data is **DO**. Use the following syntax for the item description for DO objects:

objectname[\objectname...]\dataitem

Description

- ***objectname*** is the name of the object(s) from which data is collected. You can request a number of objects in each DDE request.
- ***dataitem*** specifies the data to collected, common for all objects in the item description, and can be any of the following:

  All, Full defines all the data items that can be collected for the object.

  Value is the object data value.

  Status is a bitmap with some status information of the object.

The item description is limited to 1000 characters.

Example

If you want to obtain the values for three DO objects (DOC21_1, DOC21_2 and DOC21_3) using Microsoft Excel, use the following DDE link:

SV100|DO!DOC21_1\DOC21_2\DOC21_3\VALUE

Return value

The maximum data range returned from a DO object is as follows,

Cell address	Data cells	
A1	Status	Status of the request
A2	Value	Value of the DO

Figure F-4. Maximum Data Range Returned from a DO Object

where:

- **Value** is an long integer cell.
- **Status** is a long integer cell defined as a bitmap where each bit holds different status information:

Bit 0 = IMPLEMENTED
Bit 1 = ERROR
Bit 2 = UPDATED
Bit 3 = OUTP_BLK
Bit 4 = not used
Bit 5 = SELECTED
Bit 6 = NORMAL_OBJ_TREAT
Bit 7 = MAN_MODE
Bit 8 = VALUE
Bit 9 = not used
Bit 10 = START_VALUE
Bit 11 = not used
Bit 12 = ALARM_UNACK
Bit 13 = not used
Bit 14 = not used
Bit 15 = not used
Bit 16 = ALARM_BLK
Bit 17 = ALARM_PERIOD_BLK
Bit 18 = PRINT_BLK
Bit 19 = OUTP_RESTART
Bit 20 = LOCK
Bit 21 = INVERTED
Bit 22 = not used
Bit 23 = TESTED
Bit 24 = ORDER_TO
Bit 25 = ORDER_FROM
Bit 26 = LINKED
Bit 27 = not used
Bit 28 = not used
Bit 29 = not used
Bit 30 = not used
Bit 31 = not used

F.6 DAT Objects

Purpose

DAT objects collect calculated data from the ABB automation system.

Synopsis

The topic name used for collection of analog input data is DAT. Use the following syntax for the item description for DAT objects:

objectname[\objectname...]\dataitem

Description

- ***objectname*** is the name of the object(s) from which data is collected. You can request a number of objects in each DDE request.
- ***dataitem*** specifies the data to be collected, common to all objects in the item description, and can be any of the following:

  All, **Full** define all the data items that can be collected for the object.

  Value is the object data value.

  Valid is a Boolean flag for the validity of the DAT value.

The item description is limited to 1000 characters.

Example

To obtain all data items for three DAT objects (DAT21_1, DAT21_2 and DAT21_3) using Microsoft Excel, use the following DDE link:

SV100|DAT!DAT21_1\DAT21_2\DAT21_3\ALL

Return Value

The maximum data range returned from a DAT label is as follows,

Cell address	Data cells	
A1	Valid	Data valid/not valid
A2	Value	Value of the DAT

Figure F-5. Maximum Data Range Returned from a DAT Label

where:

- **Valid** is a long integer cell that can hold either of the following.

  0 = Not valid.

  1 = Valid.

- **Value** can be floating-point number, integer or a long integer cell.

F.7 MDAT Objects

Purpose

MDAT objects collect a number of consecutive DAT objects from the ABB automation system.

Synopsis

The topic name used for collection of Multi-DAT objects is MDAT. Use the following syntax for the item description for MDAT objects:

objectname\dataitem

Description

- ***objectname*** is the name of the first DAT object from which data is collected.
- ***dataitem*** specifies the data to be collected, and can be any of the following:

  B1..B50

  I1..I50

  L1..L50

  R1..R50

NOTE

These values represent the location of the requested DAT in the MDAT. B stands for Boolean, I for integer, IL for long integer and R for real DAT values. You can mix these items in one request in up to eight different groups, but the total number DATs requested cannot exceed 50.

Example

If you want to obtain the 30 DAT values from DAT21_1 and forward, where the first five are real values, the next 10 are Boolean, the next five are integer, the next five are Boolean and the last five are real values, use the following DDE link:

SV100|MDAT!DAT21_1\R5B10I5B5R5

Return Value

The data range returned from a MDAT object is as follows,

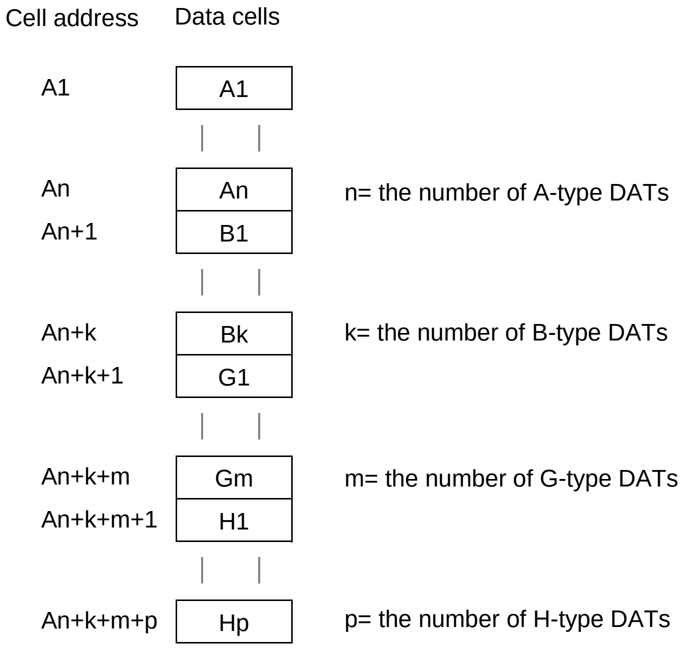

Cell address
Data cells
A1
A1
An
An
n= the number of A-type DATs
An+1
B1
An+k
Bk
k= the number of B-type DATs
An+k+1
G1
An+k+m
Gm
m= the number of G-type DATs
An+k+m+1
H1
An+k+m+p
Hp
p= the number of H-type DATs

Figure F-6. Data Range Returned from a MDAT Object

where you can group the data into as many as eight groups in such a way that **n**, **k**, **m** and **p** each can take a value in the range 0-50, but the sum of n+k+m+p cannot exceed 50. The data cells can floating-point number, integer or long integer cells.

F.8 DSC Objects

Purpose

DSC objects collect or send Data Set from/to the ABB automation system.

Synopsis

The topic name used for Data Set is DSC. Use the following syntax for the item description for DSC objects:

\netw\node\id\dataitem

Description

- ***netw*** is the network number of the source DS.
- ***node*** is the node number of the source DS.
- ***id*** is the identity number of the source DS.
- ***dataitem*** specifies the data to collected, and can be any of the following:

  B1..B24

  I1..I24

  L1..L24

  R1..R24

NOTES

- In the above values, B stands for Boolean, I for integer, L for long integer and R for real DAT values. The total number of requested DAT’s cannot exceed 24, which is the maximum number of DATs in a single DSC request.
- The order in which these data types are specified must always be: Boolean first (B), then integer (I), then long integer (L) and finally real values (R). However, one or more of these data types can be omitted such as B2L5R10, I4L5, R24, etc.

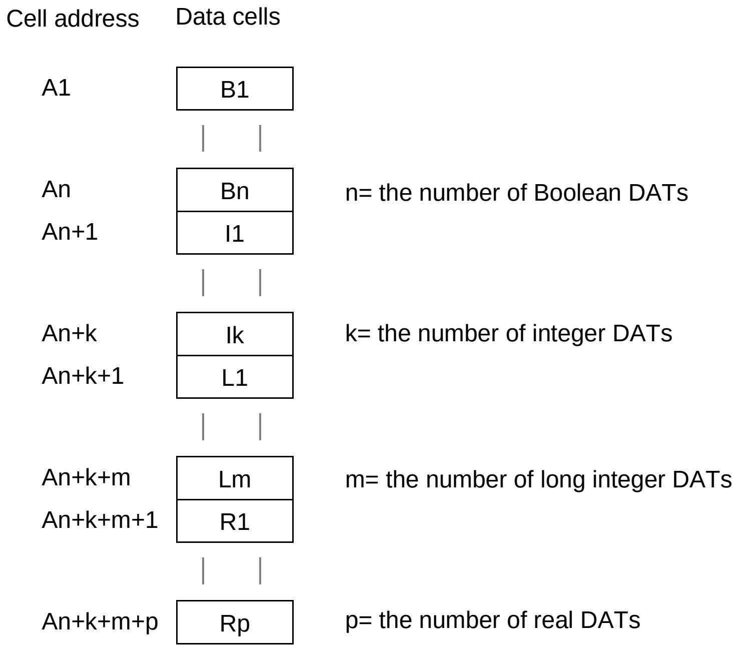
Example

If you want to obtain the values for a Data Set from network 13, node 21 with identity number 9, consisting of two Boolean values, one integer, five integer long values and seven real values, use the following DDE link:

SV100|DSC!\13\21\9\B2I1L5R7

Return Value

The data range returned from a DSC object is as follows,



Cell address	Data cells	
A1	B1	
An	Bn	n= the number of Boolean DATs
An+1	I1	
An+k	Ik	k= the number of integer DATs
An+k+1	L1	
An+k+m	Lm	m= the number of long integer DATs
An+k+m+1	R1	
An+k+m+p	Rp	p= the number of real DATs

Figure F-7. Data Range Returned from a DSC Object

where **b**, **i**, **l** and **r** can each take values 0-24, but the sum of b+i+l+r cannot exceed 24. The data cells are all floating-point number cells. The DSC data range has the same layout for both sending and receiving DSC report entries. Sending data ranges may contain integer, floating-point-number or formula cells.

F.9 TSC Objects

Purpose

TSC objects collect or send Text Set from/to the ABB automation system.

Synopsis

The topic name used for Text Set is TSC**.** Use the following syntax for the item description for TSC objects:

\netw\node\id\dataitem

Description

- ***netw*** is the network number of the source TS.
- ***node*** is the node number of the source TS.
- ***id*** is the identity number of the source TS.
- ***dataitem*** specifies the data to collected and can be any of the following:

  All, Full defines all the data items that can be collected for the object.

  Num the numeric part of the data for the TS.

  Text the text string for TS.

Example

If you want to obtain all the values for a Text Set, from network 13, node 19 with identity number 3, use the following DDE link:

SV100|TSC!\13\19\3\All

Return Value

The maximum data range returned from a TSC object is as follows,

Cell address	Data cells	
A1	Intl	Long integer value
A2	Real	Real value
A3	Bool	Boolean value
A4	String	Text string

Figure F-8. Maximum Data Range Returned from a TSC Object

where:

- **Intl** is a floating-point number cell.
- **Real** is a floating-point number cell.
- **String** is a label cell.
- **Bool** is a floating-point number cell defined as a bitmap, with each bit holding different information for presentation in the operator station (OS).

  Bit 0 = VALID2
  Bit 1 = SELECTED
  Bit 2 = MAN
  Bit 3 = USER_B
  Bit 4 = COLOUR1
  Bit 5 = COLOUR2
  Bit 6 = BLANKT
  Bit 7 = BLANKB
  Bit 8 = BLANKR
  Bit 9 = BLANKIL
  Bit 10 = B1_VAL
  Bit 11 = B2_VAL
  Bit 12 = B3_VAL
  Bit 13 = B4_VAL
  Bit 14 = `not used`
  Bit 15 = `not used`

Appendix G RFS

G.1 RFS Remote File Functions

The RFS remote functions provide a simple user interface for file transferring between two RFS nodes.

G.1.1 rfsdld

Purpose

The **rfsdld** function is the file download function.

Synopsis

MS-Windows version:

```
signed long FAR PASCAL wrfsdld(
        unsigned char     network,
        unsigned char     node,
        char FAR*         host_file,
        char FAR*         remote_file,
        unsigned char     file_format,
        signed long       delay_time);
```

DOS version:

```
signed long rfsdld(
        unsigned char     network,
        unsigned char     node,
        char*             host_file,
        char*             remote_file,
        unsigned char     file_format,
        signed long       delay_time);
```

Description

Use the rfsdld function to download any record-oriented sequential file, record by record, from the host system to the remote node over GCOM. To improve performance, send as many records as the data packets on the bus can accommodate at one time. The rfsdld function does not overwrite an existing file, while the **rfsrep** function does.

- ***network*** is a destination network (11-127). It defines the destination network number for the file.
- ***node*** is the destination node (1-127). It defines the destination node number for the file
- ***host_file*** is a NULL-terminated host file specification, POSIX-style.
- ***remote_file*** is a NULL-terminated host file specification, POSIX-style.
- ***file_format*** is the file format. It can have one of the values *rfs_binary_file* or *rfs_ascii_file*.
- ***delay_time*** is the delay time (in msec.) between request messages. It is used to control the load on the network.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It can have of the values *rfs_eof* or *rfs_success.* For other error codes refer to <errno.h> which is a standard C include file**.**

G.1.2 rfsrep

Purpose

The **rfsrep** function is the file replace function.

Synopsis

MS-Windows version:

```
signed long FAR PASCAL wrfsrep (
        unsigned char   network,
        unsigned char   node,
        char FAR*       host_file,
        char FAR*       remote_file,
        unsigned char   file_format,
        signed long     delay_time);
```

DOS version:

```
signed long rfsrep (
        unsigned char   network,
        unsigned char   node,
        char*           host_file,
        char*           remote_file,
        unsigned char   file_format,
        signed long     delay_time);
```

Description

Use the rfsrep function to overwrite a record-oriented sequential file in the remote node, record by record, with a file from the host system over GCOM. If the remote node file does not exist, rfsrep creates it. To improve performance, send as many records as the bus packets can accommodate at one time. The rfsrep function overwrites an existing file, while the **rfsdld** function does not.

- ***network*** is the destination network (11-127). It defines the destination network number for the file.
- ***node*** is the destination node (1-127). It defines the destination node number for the file.
- ***host_file*** is the NULL-terminated host file specification, POSIX-style.
- ***remote_file*** is the NULL-terminated host file specification, POSIX-style.
- ***file_forma*** is the file format. It can have one of the values, *rfs_binary_file* or *rfs_ascii_file*.
- ***delay_time*** is the delay time (in msec.) between request messages. It is used to control the load on the network.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have the values *rfs_eof* or *rfs_success*. For other error codes refer to <errno.h> which is a standard C include file**.**

G.1.3 rfsuld

Purpose

The **rfsuld** function is the file upload function.

Synopsis

MS-Windows version:

```
signed long FAR PASCAL wrfsuld (
        unsigned char    network,
        unsigned char    node,
        char FAR*        host_file,
        char FAR*        remote_file,
        unsigned char    file_format,
        signed long      delay_time);
```

DOS version:

```
signed long rfsuld (
        unsigned char    network,
        unsigned char    node,
        char*            host_file,
        char*            remote_file,
        unsigned char    file_format,
        signed long      delay_time);
```

Description

Use the rfsuld function to transfer any record-oriented sequential file, record by record, from a remote node to the host system over GCOM. To improve performance, have as many records read at one time as the bus packets can accommodate. The rfsuld function does not overwrite an existing file.

- ***network*** is the destination network (11-127). It defines the destination network number for the file.
- ***node*** is the destination node (1-127). It defines the destination node number for the file.
- ***host_file*** is the NULL-terminated host file specification, POSIX-style.
- ***remote_file*** is the NULL-terminated host file specification, POSIX-style.
- ***file_format*** is the file format. It can have one of the values, *rfs_binary_file* or *rfs_ascii_file*.
- ***delay_time*** is the delay time (in msec.) between request messages. It is used to control the load on the network.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file.

G.1.4 rfsdel

Purpose

The **rfsdel** function is the file delete function.

Synopsis

MS-Windows version:

```
signed long FAR PASCAL wrfsdel (
        unsigned char    network,
        unsigned char    node,
        char FAR*        remote_file);
```

DOS version:

```
signed long rfsdel (
        unsigned char    network,
        unsigned char    node,
        char*            remote_file);
```

Description

Use the rfsdel function to delete a file in a remote node.

- ***network*** is the destination network (11-127). It defines the destination network number for the file.
- ***node*** is the destination node (1-127). It defines the destination node number for the file
- ***remote file*** is a NULL-terminated host file specification, POSIX-style.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file.

G.1.5 rfsrnm

Purpose

The **rfsrnm** function is the file rename function.

Synopsis

MS-Windows version:

```
signed long FAR PASCAL wrfsrnm (
        unsigned char   network,
        unsigned char   node,
        char FAR*       old_file_name,
        char FAR*       new_file_name);
```

DOS version:

```
signed long rfsrnm (
        unsigned char   network,
        unsigned char   node,
        char*           old_file_name,
        char*           new_file_name);
```

Description

Use the rfsrnm function to rename a file in a remote node. The rfsrnm function overwrites an existing file.

- ***network*** is the destination network (11-127). It defines the destination network number for the file.
- ***node*** is the destination node (1-127). It defines the destination node number for the file
- ***old_file_name*** is a NULL-terminated host file specification, POSIX-style.
- ***new_file_name*** is a NULL-terminated host file specification, POSIX-style.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file**.**

G.1.6 rfstra

Purpose

The **rfstra** function is the file translate function.

Synopsis

MS-Windows version:

```
signed long FAR PASCAL wrfstra (
        unsigned char              network,
        unsigned char              node,
        char FAR*                  report name,
        char FAR*                  unit_name,
        char FAR*                  dir_apth,
        rfs_time_type FAR*         start_time,
        rfs_time_type FAR*         end_time,
        unsigned long              index,
        unsigned long FAR*         trans_status,
        unsigned long FAR*         num_reports,
        char*                      file_path_list,
        unsigned short FAR*        year_list,
        unsigned char FAR*         month_list,
        unsigned char FAR*         day_list,
        unsigned char FAR*         hour_list,
        unsigned char FAR*         min_list,
        unsigned char FAR*         sec_list);
```

DOS version:

```
signed long rfstra (
        unsigned char              network,
        unsigned char              node,
        char*                      report name,
        char*                      unit_name,
        char*                      dir_apth,
        rfs_time_type*             start_time,
        rfs_time_type*             end_time,
        unsigned long              index,
        unsigned long*             trans_status,
        unsigned long*             num_reports,
        char*                      file_path_list,
        unsigned short*            year_list,
        unsigned char*             month_list,
        unsigned char*             day_list,
        unsigned char*             hour_list,
        unsigned char*             min_list,
        unsigned char*             sec_list);
```

Description

Use the rfstra function to translate a logical report name to a physical report name(s) in a MOD 300 system, based on the parameters passed to the function. You can then transfer the report file(s) to the host by means of the standard remote file transfer function, **rfsuld**.

- ***network*** is the destination network (11-127). It defines the destination network number for the file.
- ***node*** is the destination node (1-127). It defines the destination node number for the file.
- ***report_name*** is a NULL-terminated logical report name.
- ***unit_name*** is a NULL-terminated host file specification, POSIX-style.
- ***dir_path*** is a file format. It can have one of the values, *rfs_binary_file* or *rfs_ascii_file*.
- ***start_time*** is a rough start time for the report.
- ***end_time*** is a rough end time for the report.
- ***index*** is the Nth report desired.
- ***trans_status*** is the status of the translation, which can be one of the following values:

  rfs_ra_t_no_err = No error value returned

  rfs_ra_t_no_rep = No such report

  rfs_ra_t_no_ins = No such report instance

  rfs_ra_t_rep_dir_err = Error in report directory (see Table 6-8)

  rfs_ra_t_arch_dir_err = Error in archive directory (see Table 6-8)

  rfs_ra_t_time_range_err = No reports in this time range

  rfs_ra_t_index_err = Index value too large

  rfs_ra_t_eof = Reached the end of the file

  rfs_ra_t_arch_dev_busy = Archive device currently busy

  num_reports = Number of reports created
- ***file_path_list*** is a full list of null-terminated physical report names, POSIX-style. The last report file in the list has the latest date. Array (**RFS_MAX_REPORT**).
- ***year_list*** is a list containing the years corresponding to the physical report names.
- ***month_list*** is a list containing the months corresponding to the physical report names.
- ***day_list*** is a list containing the days corresponding to the physical report names.
- ***hour_list*** is a list containing the hours corresponding to the physical report names.
- ***min_list*** is a list containing the minutes corresponding to the physical report names.
- ***sec_list*** is a list containing the seconds corresponding to the physical report names.

The following relationship exists between the start time, the end time and the index parameters:

Table G-1. Start Time, End Time and Index Parameter Relationship in rfstra

Start_time end_time	**Index**	**Summary**
0	0	Return the first report
0	non-zero	Return the Nth - 1 report
non-zero	0	Return all reports between the specified times
non-zero	non-zero	Return all reports between the specified times, starting with Nth - 1 report

If you define the parameters according to the last two relationships, you must call the rfstra function from inside a loop by specifying the appropriate start and end times until all the report file names are received.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file**.**

G.1.7 rfstcl

Purpose

The **rfstcl** function is the Taylor Control Language function.

Synopsis

MS-Windows version:

```
signed long FAR PASCAL wrfstcl (
        unsigned char          network,
        unsigned char          node,
        char FAR*              file_path,
        unsigned long          op_code,
        unsigned long          tcl_type,
        rfs_tcl_file_descrip   file_descrip);
```

DOS version:

```
signed long rfstcl (
        unsigned char          network,
        unsigned char          node,
        char*                  file_path,
        unsigned long          op_code,
        unsigned long          tcl_type,
        rfs_tcl_file_descrip   file_descrip);
```

Description

The Taylor Control Language Catalog, rfstcl, function allows applications residing in the host computer environment to execute a set of management functions on TCL catalogs for both recipes and sequences in MOD 300. The available management functions are:

- Create the necessary files and entries in the appropriate catalog. (The source must exist when this request is made.)
- Compile a sequence or recipe. (Execute a create function before compilation is requested.)
- Compile a sequence or recipe and link the file if compilation is successful.
- Link the (compiled) file.
- Delete the sequence or recipe and the appropriate entries from the corresponding TCL catalog displays.

Whenever a TCL program or recipe is initially built in the host system, it may be transferred to the MOD 300 system by one of the standard remote file transfer functions, rfsdld or rfsrep. Use rfsuld function to retrieve TCL programs or recipes from the MOD 300 system.

- ***network*** is a destination network (11-127). It defines the destination network number where the sequence or receipt resides.
- ***node*** is the destination node (1-127). It defines the destination node number where the sequence or receipt resides.
- ***file path*** is a NULL-terminated remote file specification, POSIX-style.
- ***op_code*** is an operation code indicating the action to take and can have the following values:

    rfs_tcl_sched = TCL scheduler routine

    rfs_tcl_proc = TCL procedure routine

    rfs_tcl_oper = TCL operations routine

    rfs_tcl_recipe = TCL recipe file

    rfs_tcl_incl = TCL include file

- ***tcl_type*** is the file format. It can have one of the values, *rfs_binary_file* or *rfs_ascii_file*.
- ***file_descrip*** is the delay time (in msec) between request messages. Use it to control the load on the network.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file.

G.2 RFS Local File Primitives

Use the local file primitives instead of standard file-handling routines to free yourself from the details of description file-handling. You only need these routines if a specific record structure is vital for a file that is not created by the RFS server or the RFSFTU program.

G.2.1 rfslop

Purpose

The **rfslop** function is the local file open function.

Synopsis

MS-Windows version:

```
signed long FAR PASCAL wrfslop (
        char FAR*        file_path,
        unsigned char    access_type,
        unsigned char    file_format,
        void FAR**       file_ptr);
```

DOS version:

```
signed long rfslop (
        char*            file_path,
        unsigned char    access_type,
        unsigned char    file_format,
        void**           file_ptr);
```

Description

Use the rfslop function to open a file in the local file system for processing, according to the access type specified. Access types are **Read** and **Read-and-write** (general file access). The rfslop function returns a structure that is associated with the file. This pointer is used for Read-and-write access and for Close commands.

- ***file_path*** is a NULL-terminated file specification, POSIX-style.
- ***access_type*** is the file access type, which can be one of the following values:

  rfs_read = Read-only permission

  rfs_read_and_write = Read-and-write permission
- ***file_format*** is the file format. It can have one the values *rfs_binary_file* or *rfs_ascii_file*.
- ***file_ptr*** is the address of the structure associated with the file.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file**.**

G.2.2 rfslcl

Purpose

The **rfslcl** function is the local file close function.

Synopsis

MS-Windows version:

```
signed long FAR PASCAL wrfslcl (
        void FAR*    file_ptr);
```

DOS version:

```
signed long rfslcl (
         void*          file_ptr);
```

Description

The rfslcl function closes a file in the local file system.

- ***file_ptr*** is the address of the structure associated with the file.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file**.**

G.2.3 rfsdel

Purpose

The **rfsdel** function is the local file delete function.

Synopsis

MS-Windows version:

```
signed long FAR PASCAL wrfsdel (
        char FAR*    file_path);
```

DOS version:

```
signed long rfsdel (
        char*        file_path);
```

Description

The **rfsdel** function deletes a file in the local file system.

- ***file_path*** is a NULL-terminated file specification, POSIX-style.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file**.**

G.2.4 rfslrn

Purpose

The **rfslrn** function is the local file rename function.

Synopsis

MS-Windows version:

signed long FAR PASCAL wrfslrn (
char FAR* *old_file_path,*
char FAR* *new_file_path);*

DOS version:

signed long rfslrn (
char* *old_file_path,*
char* *new_file_path);*

Description

The rfslrn function changes the name of a file in the local file system. The existing file is overwritten.

- ***old_file_path*** is a NULL-terminated old file specification, POSIX-style.
- ***new_file_path*** is a NULL-terminated new file specification, POSIX-style.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file**.**

G.2.5 rfslcr

Purpose

The **rfsclr** function is the local file create function.

Synopsis

MS-Windows version:

```
signed long FAR PSCAL wrfslcr (
        char FAR*       file_path,
        size_t          rec_length,
        unsigned short  file_mode);
```

DOS version:

```
signed long rfslcr (
        char*           file_path,
        size_t          rec_length,
        unsigned short  file_mode);
```

Description

Create a sequential file in the local file system with the rfslcr function. No space is allocated on the media at creation, but the space is dynamically allocated by writes. In files with variable file-length records, the record length is set to zero. **File mode** is a value specifying **READ** and **WRITE** protection for the file (please refer to C standard file creation). The rfslcr function also creates a description file via an implicit call to the
rfslcd function.

- ***file_path*** is a NULL-terminated file specification, POSIX-style.
- ***rec_length*** is the record size. If the records are to be of variable length, the value zero is used.
- ***file_mode*** is the file protection mode, POSIX-style.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file**.**

G.2.6 rfslgr

Purpose

The **rfslgr** is the local file read function.

Synopsis

MS-Windows version:

```
signed long FAR PASCAL wrfslgr (
        char FAR*               file_ptr,
        size_t                  buffer_size,
        size_t                  record_length,
        unsigned char FAR*      buffer);
```

DOS version:

```
signed long rfslgr (
        char*            file_ptr,
        size_t           buffer_size,
        size_t           record_length,
        unsigned char*   buffer);
```

Description

The rfslgr function reads a record from a sequential file in a user-specified buffer. The file resides in the local file system.

- ***file_ptr*** is the address of a structure associated with the file.
- ***buffer_size*** is the size of the data-read buffer in bytes.
- ***record_length*** is the record size in bytes.
- ***buffer*** is the data read buffer.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file.

G.2.7 rfslpr

Purpose

The **rfslpr** function is the local file write function.

Synopsis

MS-Windows version:

```
signed long FAR PACSAL wrfslpr (
        char FAR*             file_ptr,
        size_t                record_length,
        unsigned char FAR*    buffer);
```

DOS version:

```
signed long rfslpr (
        char*             file_ptr,
        size_t            record_length,
        unsigned char*    buffer);
```

Description

The rfslpr function puts a record into a sequential file from a user-specified buffer. The file resides in the local file system.

- ***file_ptr*** is the address of a structure associated with the file.
- ***record_length*** is the record size, in bytes.
- ***buffer*** is the data write buffer.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file**.**

G.2.8 rfslsp

Purpose

The **rfslsp** function is the local file position function.

Synopsis

MS-Windows version:

signed long FAR PASCAL wrfslsp (
char FAR* *file_ptr);*

DOS version:

signed long rfslsp (
char* *file_ptr);*

Description

The rfslsp function sets the file position to the last record. The file resides in the local file system.

- ***file_ptr*** is the address of a structure associated with the file.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file**.**

G.2.9 rfslgf

Purpose

The **rfslgf** function is the local file get attribute function.

Synopsis

MS-Windows version:

```
signed long FAR PASCAL wrfslgf (
        char FAR*             file_ptr,
        rfs_file_attr FAR*    file_attr);
```

DOS version:

```
signed long rfslgf (
        char*            file_ptr,
        rfs_file_attr*   file_attr);
```

Description

The rfslgf function reads the file attributes for a local file (using *file_ptr)* and writes the information to the user-specified *file_attr* buffer.

- ***file_ptr*** is the address of a structure associated with the file.
- ***file_attr*** is a buffer containing the file attributes.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file**.**

G.2.10 rfslcd

Purpose

The **rfslcd** function is the local file create function.

Synopsis

MS-Windows version:

```
signed long FAR PASCAL wrfslcd (
        char FAR*          file_path,
        unsigned char      rec_format,
        unsigned short     rec_size);
```

DOS version:

```
signed long rfslcd (
        char*              file_path,
        unsigned char      rec_format,
        unsigned short     rec_size);
```

Description

The rfslcd function creates a description file. The file resides in the local file system. Call this function for files not created with the rfslcr function.

- ***file_path*** is a NULL-terminated file specification, POSIX-style.
- ***rec_format*** is the record format, which can have one of the following values:

  variable = Variable record format

  fixed = Fixed record format
- ***rec_size*** is the record size. If the records are of variable length, the value zero is used.

Return Value

A POSIX-style error code indicates whether the function is successful or not. It could have one of the values, *rfs_eof* or *rfs_success*. For other error codes, refer to <errno.h> which is a standard C include file**.**

G.3 RFS Error Log Function

You can use the error log function to get more detailed information about errors that appear.

G.3.1 rfselr

Purpose

The **rfselr** is the local error function.

Synopsis

MS-Windows version:

signed long FAR PASCAL wrfselr (
char FAR* *error_mess);*

DOS version:

signed long rfselr (
char* *error_mess);*

Description

The rfselr function reads error-log messages from the RFS library functions. You can call the rfselr function when an RFS function (rfsdld, rfsrep, etc.) fails. Error-log messages are ASCII strings, specifying the function that failed in greater detail. These error messages are listed in Section 6.1.3.7, Error Messages from the RFSFTU Program.

- ***error_mess*** is an error-log message from an RFS library function.

Return Value

A POSIX-style error code indicates whether the function is successful or not.

G.4 RFS DLL Functions (MS-Windows only)

Use the RFS connect/disconnect functions in user applications in MS-Windows to properly connect/disconnect to the RFSAPI.DLL. These functions, connect/disconnect, should be the first and last functions a user application calls in the RFSAPI.DLL.

G.4.1 wrfscon

Purpose

The **wrfscon** function is the RFSAPI.DLL connect function.

Synopsis

MS-Windows version:

```
bool FAR PASCAL wrfscon (
      HANDLE      hwnd);
```

Description

Use the wrfscon function to properly connect to RFSAPI.DLL. You must connect before you can use any of the other RFS functions.

- ***hwnd*** is the window handle for the user application

Return Value

A Boolean value indicates whether the function is successful or not.

G.4.2 wrfsdis

Purpose

The **wrfsdis** function is the RFSAPI.DLL disconnect function.

Synopsis

MS-Windows version:

bool FAR PASCAL wrfsdis (
HANDLE *hwnd);*

Description

Use the **wrfsdis** to properly disconnect from the RFSAPI.DLL when the RFS application terminates.

- ***hwnd*** is the window handle for the user application

Return Value

A Boolean value indicates whether the function is successful or not.

G.4.3 wrfscnf

Purpose

The **wrfscnf** function displays a dialog box for RFS configuration.

Synopsis

MS-Windows version:

void FAR PASCAL wrfscnf (
HANDLE *hwnd);*

Description

Use the **wrfscnf** to modify the RFS communication parameters. For more information please refer to Section 4.3.10 RFS Configuration.

- ***hwnd*** is the window handle for the user application

Return Value

No return value.

G.5 Creating an RFS Application

G.5.1 C-type Declarations

```
#define RFS_FILE_PATH_LEN      200
#define RFS_REP_NAME_LEN       13
#define RFS_UNIT_NAME_LEN      13
#define RFS_MAX_REPORT         7
#define RFS_TCL_FILE_DESCR_LEN 20

/* Error codes. For additional error codes, see the POSIX manual
*/ /*
==============================================================
*/ #define RFS_EOF    -1      /* End Of File reached    */
#define RFS_SUCCESS   0       /* Function was successful */

/* File format types */
/* ================= */
#define RFS_BINARY_FILE      0
#define RFS_ASCII_FILE       1

/* File access level */
/* ================= */
#define RFS_READ_ONLY        0
#define RFS_READ_FOR_UPDATE  1
#define RFS_READ_AND_WRITE   2

/* RFSTRA trans_status return values */
/* ================================= */
#define RFS_RA_T_NO_ERR      0     /* No error return value*/
#define RFS_RA_T_NO_REP      1     /* No such report*/
#define RFS_RA_T_NO_INS      2     /* No such report instance*/
#define RFS_RA_T_REP_DIR_ERR 3     /* Error in report dir*/ /*
(see diag.)*/
#define RFS_RA_T_ARCH_DIR_ERR 4    /* Error in archive dir */ /*
(see diag.)*/
#define RFS_RA_T_TIME_RANGE_ERR5   /* No reports in time
range*/
#define RFS_RA_T_INDEX_ERR   6     /* Index value too large*/
#define RFS_RA_T_EOF         7     /* Reached end of file*/
 #define RFS_RA_T_ARCH_DEV_BUSY8   /* Archive device cur-
rently*/ /* busy */

/* TCL operation codes */
/* =================== */
#define RFS_TCL_CREATE       20
#define RFS_TCL_COMP_REQ     1
#define RFS_TCL_COMPL_REQ    10
#define RFS_TCL_LINK         19
#define RFS_TCL_DELETE       21
```

```
/* TCL file types */
/* ============== */
#define RFS_TCL_SCHED        269
#define RFS_TCL_PROC         270
#define RFS_TCL_OPER         271
#define RFS_TCL_RECIPE       388
#define RFS_TCL_INCL         402

/* File attributes definition */
#define RFS_VAR_REC          0
#define RFS_FIX_REC          1

typedef struct

            unsigned char   rfm;
            unsigned short  mrs;
        rfs_file_attr;
```

G.5.2 Compile and Link

MS-Windows Version

C-type declarations are used by files including the RFSWIN.H file in the user program. The RFSAPI.DLL is linked during runtime, and is not used during the application building. You can use any compiler and linker you choose.

DOS Version

C-type declarations are used by files including the RFS.H file in the user program. User programs are linked to the Remote File Handling Library using the RFS.LIB library file. Compile the user programs based on the large memory model and link the program based on a stack size of 12 Kbytes.

NOTE

Use the Microsoft C run-time library when linking the DOS version of the RFS library.

G.5.3 C-code Example

This section gives an example of a C-code of an RFS application in DOS, using the RFS file transfer library.

Example

```
/*************************************************************/
/* This C program shows an example of how to call the Remote */
/* FileHandling Library functions RFSDLD, RFSREP, RFSULD,    */
/* RFSTRA and RFSTCL. The parameters needed are read         */
/* interactively. In this example, the start and end times   */
/* for the RFSTRA function must be given as non-zero values. */
/* The report file names are read in a loop until all file   */
/* names in the specified time interval have been received.  */

#include <stdio.h>   /* Standard I/O definitions */
#include <errno.h>   /* POSIX-style error code definitions */
#include "rfs.h"   /* RFS library function definitions */

int main (void)

{

 char parameter[255];
 int selection;
 unsigned char rep;

 /* Parameters for all functions */
 unsigned char network;
 unsigned char node;
 long status;

 /* Parameters for the RFSDLD, RFSREP, RFSULD functions */
 char host_file[RFS_FILE_PATH_LEN];
 char remote_file[RFS_FILE_PATH_LEN];
 unsigned char file_format;
 unsigned short delay_time;

 /* Parameters for the RFSTRA */
 char report_name[RFS_REP_NAME_LEN];
 char dir_path[RFS_FILE_PATH__LEN];
 char unit_name[RFS_UNIT_NAME_LEN];
 rfs_time_type start_time;
 rfs_time_type end_time;
 long index;
 long trans_status;
 long num_reports;
 char file_path_list[RFS_MAX_REPORT][RFS_FILE_PATH_LEN];
 unsigned shortyear_list[RFS_MAX_REPORT];
 unsigned char month_list[RFS_MAX_REPORT];
 unsigned char day_list[RFS_MAX_REPORT];
 unsigned char hour_list[RFS_MAX_REPORT];
 unsigned char min_list[RFS_MAX_REPORT];
 unsigned char sec_list[RFS_MAX_REPORT];

 /* Parameters for the RFSTCL function */
 char file_path[RFS_FILE_PATH_LEN];
 long op_code;
 long tcl_type;
 char file_descr[RFS_TCL_FILE_DESCR_LEN];

 printf ("\n+---------------------------------------+\n");
 printf ("! Remote File System, C program example !\n");
 printf ("+---------------------------------------+\n\n");

 printf ("1 - Download a sequential file to a remote node\n");
 printf ("2 - Replace a sequential file in a remote node\n");
 printf ("3 - Upload a sequential file from a remote node\n");
 printf ("4 - Translate a logical report name to physical report
                   name(s)\n");
 printf ("5 - Perform MOD 300 TCL catalog function\n\n");
 printf ("Enter selection: ");
 gets (parameter);
 selection = atoi (parameter);


 printf ("\n\n");

 if ((selection >= 1) && (selection <= 5))
{
  printf ("Destination network (11..127)     : ");
  gets (parameter);
  network = atoi (parameter);
  printf ("Destination node (1..127)     : ");
  gets (parameter);
  node = atoi (parameter);

  if ((selection >= 1) && (selection <= 3))
  {
    printf ("Host file specification, POSIX-style  : ");
    gets (host_file);
    printf ("Remote file specification, POSIX-style: ");
    gets (remote_file);
    printf ("File format (0=binary, 1=ASCII)       : ");
    gets (parameter);
    file_format = atoi (parameter);
    printf ("Delay time in msec       : ");
    gets (parameter);
    delay_time = atoi (parameter);
  } /* then */

  switch (selection)
  {
    case 1: printf ("\n\nDownloading a sequential file...\n");
            status = rfsdld (network, node, host_file,
                         remote_file, file_format,
                         delay_time);
        if (status != RFS_SUCCESS)
          printf ("RFSDLD failure, return status: %d\n",
                   status);
        break;
case 2: printf ("nnReplacing a sequential file...n");
        status = rfsrep (network, node, host_file,
                         remote_file, file_format,
                       delay_time);
        if (status != RFS_SUCCESS)
        printf ("RFSREP failure, return status: %d\n",
                 status);
        break;
case 3: printf ("\n\nUploading a sequential file...\n");
        status = rfsuld (network, node, host_file,
                         remote_file, file_format,
                         delay_time);
        if (status != RFS_SUCCESS)
          printf ("RFSULD failure, return status: %d\n",
                   status);
        break;
case 4: printf ("Logical report name       : ");
        gets (report_name);
        printf ("Unit name       : ");
        gets (unit_name);
        printf("Remote directory path in POSIX style   : ");
        gets (dir_path);
        printf("Enter start and end times as non-zero
                values\n");
        printf ("Start time, month (1-12)        : ");
        gets (parameter);
        start_time.month = atoi (parameter);
        printf ("    day (1-31)       : ");
        gets (parameter);
        start_time.day = atoi (parameter);
        printf ("    year (1970-9999)       : ");
        gets (parameter);
        start_time.year = atoi (parameter);
        printf ("    hour (0-23)       : ");
        gets (parameter);
        start_time.hour = atoi (parameter);
        printf ("    minute (0-59)       : ");
        gets (parameter);
        start_time.minute = atoi (parameter);
        printf ("    second (0-59)       : ");
        gets (parameter);
        start_time.sec = atoi (parameter);
        printf ("End time, month (1-12)       : ");
```

```
gets (parameter);
end_time.month = atoi (parameter);
printf ("  day (1-31)       : ");
gets (parameter);
end_time.day = atoi (parameter);
printf ("  year (1970-9999)       : ");
gets (parameter);
end_time.year = atoi (parameter);
printf ("  hour (0-23)       : ");
gets (parameter);
end_time.hour = atoi (parameter);
printf ("  minute (0-59)       : ");
gets (parameter);
end_time.minute = atoi (parameter);
printf ("  second (0-59)       : ");
gets (parameter);
end_time.sec = atoi (parameter);
printf ("Index       : ");
gets (parameter);
index = atoi (parameter);
printf ("\n\nTranslating a MOD 300 logical report
        name...\n");
do
{
  status = rfstra (network, node, report_name,
                   unit_name, dir_path, &start_time,
                   &end_time, index, &trans_status,
                   &num_reports, file_path_list,
                   year_list, month_list, day_list,
                   hour_list, min_list, sec_list);
  if (status != RFS_SUCCESS)
  {
    printf ("RFSTRA failure, return status: %dn",
             status);
    return; /* if */
  }

  if (trans_status != RFS_RA_T_NO_ERR)
  {
     printf ("RFSTRA failure, translation status:
              %d",  trans_status);
    return; /* if */
  }

  if (num_reports <= 0)
  {
    printf ("RFSTRA failure, number of report files:
            %d", num_reports);
    return; /* if */
  }
```

```
          for (rep = 0; rep < num_reports; rep++)
          {
             printf ("Physical report file <%d>\n", rep);
             printf ("    Physical report file name: %s\n",
                      &file_path_list[rep][0]);
             printf ("    Year: %d\n", year_list[rep]);
             printf ("    Month: %d\n", month_list[rep]);
             printf ("    Day: %d\n", day_list[rep]);
             printf ("    Hour: %d\n", hour_list[rep]);
             printf ("    Minute: %d\n", min_list[rep]);
             printf ("    Second: %d\n", sec_list[rep]);
          } /* for */

          /* Adjust start time */
          rep--;
          start_time.month = month_list[rep];
          start_time.day = day_list[rep];
          start_time.year = year_list[rep];
          start_time.hour = hour_list[rep];
          start_time.minute = min_list[rep];
          start_time.sec = sec_list[rep] + 1;
        } while (trans_status == RFS_RA_T_NO_ERR);
        break;

case 5: printf ("Remote file specification, POSIX-style: ");
        gets (file_path);
        printf ("Operation code       : ");
        gets (parameter);
        op_code = atoi (parameter);
        printf ("TCL file type       : ");
        gets (parameter);
        tcl_type = atoi (parameter);
        if ((op_code == RFS_TCL_CREATE) &&
        {
          (tcl_type == RFS_TCL_RECIPE))
          printf ("File descriptor : ");
          gets (file_descr);
        } /* then */
        else
          file_descr[0] = ’\0’;


        printf ("\n\nPerforming a MOD 300 TCL catalog
                 function...\n");
        status = rfstcl (network, node, file_path, op_code,
                         tcl_type, file_descr);
        if (status != RFS_SUCCESS)
          printf ("RFSTCL failure, return status: %dn",
                  status);
        break;
```

```
    default: ;
    } /* switch */
  } /* then */
}
```

INDEX

A
ABSWAIT E-1
Address 2-1
Advant 1-1
AI Label D-9
AI Objects F-2
AO Label D-11
AO Objects F-4
AUTOEXEC.BAT 2-6

B
Batch File Execution 4-10
BINDINGS 2-8
BUFFERSIZE 2-8

C
Capacity 3-20
COM Label D-28
Command Line Interface 4-6
Communication board 2-3
Compile G-27
CONFIG.SYS 2-7
Configuration 2-1
Configuration Rules A-1
Control Network A-1
CTRLNETW 2-2, 2-6
Cyclic Report Requests C-4

D
DAT Label D-17
DAT Objects F-10
Data Ranges D-1
DATE Label D-2
DDE
  Access Data 3-34
  Configuration 4-16
  Send data 3-38
  Syntax F-1
DDE Interface 4-7
DEVICE 2-7
DI Label D-13
DI Objects F-6
Disconnect Qualifier 4-6, C-2
Disk Storage 3-24
DO Label D-15
DO Objects F-8
DRIVERNAME 2-8
DSC Label D-4
DSC Objects F-13

E
Equipment requirements
  Application software 1-6
  Basic hardware 1-6
  Operating system 1-6
Error
  Labels 5-1
  Messages 5-1, C-10, C-11, C-12
EVL Label D-31
EVLC Label D-34
EXEC E-3
Execution
  Batch File 4-10
  Report 4-12
EXIT E-11

F
Formula 3-18, 3-31

G
GCOM
  Configuration 4-15
  Network Connection 4-17
  Setup File B-2
  Status 4-16
GCOM Driver 1-10
GCOMDESC.DTA B-1
GCOMSYMB.DTA B-1, B-4
GCOMSYMB.ERR B-4
GCOMSYMB.OLD B-4
GOSUB E-7
GOTO E-5

H
Help 4-17

I
IEEE 802.3 2-3
IF ERROR E-10
IF TRIGGIDENT E-9
Interface
  Command Line 4-6

  - DDE 4-7
  - Windows 4-2
- Item F-1

L
- Labels 3-4
- Link G-27

M
- Macros 3-19
- MDAT Label D-18
- MDAT Objects F-11
- Memory 3-22

N
- NDIS 1-10
- NETW 2-1, 2-6
- NODE 2-1, 2-6
- Noerror Qualifier C-3

O
- Outfile Qualifier 4-7, C-3

P
- Performance 3-20
- Plant Network A-3
- Primary Memory 3-22
- PROTMAN.DOS 2-3, 2-7
- PROTOCOL.INI 2-8

Q
- Qualifier
  - Disconnect 4-6, C-2
  - Noerror C-3
  - Outfile 4-7, C-3
  - Resolve 4-7, C-2
  - Status 4-6, C-2

R
- RELWAIT E-2
- Remote File System 1-1
- Report
  - Data Files 3-6
  - Design 3-14
  - Execution 4-12
  - Generation Strategies 3-15
  - Labels 3-4
  - Macros 3-19
  - Output 3-9
  - Planning 3-14
  - Production 3-2
  - Request 3-6
  - Scheduling 3-32
- Resolve Qualifier 4-7, C-2
- RETURN E-8
- RFS
  - Configuration 4-20
  - Configuration File B-7
  - Configuration Program C-8
  - Description Files B-6
  - DLL Functions G-24
  - Error Log Function G-23
  - File Transfer Utility Program C-6
  - Local File Primitives G-13
  - Performance 3-22
  - Remote File Functions G-1
- RFS File Transfer Library 1-9, 3-13, 4-9
- RFS Server 1-9, 4-8
- RFS_CONF 2-6
- RFSCNF C-8
- rfsdel G-6, G-15
- rfsdld G-1
- rfselr G-23
- RFSFTU C-6
- rfslcd G-22
- rfslcl G-14
- rfslcr G-17
- rfslgf G-21
- rfslgr G-18
- rfslop G-13
- rfslpr G-19
- rfslrn G-16
- rfslsp G-20
- rfsrep G-3
- rfsrnm G-7
- rfstcl G-11
- rfstra G-8
- rfsuld G-5

S
- Sending Data
  - Using DDE 3-38
  - Using Reports 3-10
- Spreadsheet Software 3-3
- Status Qualifier 4-6, C-2
- SuperView 100 1-1, 1-7
- SuperView 100 Report Software 4-1, C-1

SV 100
    Version 4-18
SV_RFSDIR 2-6
SV100 E-4
    Address 2-1
    Basic Software 1-10
    Identification 2-2
    Performance 3-20
    Report Software 1-8
SV100_ID 2-6
SV100-Distribution Files 2-3
SV100WIN.INI B-1
SVFILES 2-6
Symbol Files B-4
Syntax
    Labels D-1

T
TIME Label D-3
Topic F-1
Transit Nodes 2-2
TRIGG Label D-35
TRNODE 2-2, 2-6
TSC Label D-7
TSC Objects F-15
TTD Label D-20
TTD Symbol Files B-5
TTDO Label D-24
TTDSYMB.DTA B-1
TTDSYMB.ERR B-5
TTDSYMB.OLD B-5
TYP_DATA_LINK B-3

V
Version 4-18

X
XLS C-1

W
Windows Interface 4-2
WK1 C-1
WKS C-1
WR1 C-1
wrfscnf G-25
wrfscon G-24
wrfsdel G-6, G-15
wrfsdis G-25
wrfsdld G-1
wrfselr G-23
wrfslcd G-22
wrfslcl G-14
wrfslcr G-17
wrfslgf G-21
wrfslgr G-18
wrfslop G-13
wrfslpr G-19
wrfslrn G-16
wrfslsp G-20
wrfsrep G-3
wrfsrnm G-7
wrfstcl G-11
wrfstra G-8
wrfsuld G-5

3BSE 000 509R0101

February, 1997